Novel Block-Based Motion Estimation and Segmentation for Video Coding

Chiew Tuan Kiang

August 2004

A thesis submitted to the University of Bristol in accordance with the requirements of the degree of Doctor of Philosophy in the Faculty of Engineering

Abstract

This thesis concerns the development of motion estimation and segmentation techniques for current and near-future video compression systems. The prevailing block-matching algorithm finds the closest match between pixels in a current block with a block of pixels located in a previous frame within a neighbourhood of current blocks' location. The matching criterion commonly used is the sum-of-absolute-difference (SAD) and the motion vector of the current block is the offset from the current block which gives the minimum SAD. Although BMA via SAD minimization is very simple to implement and can readily be implemented on real-time embedded systems, it is not without its drawbacks. The two main ones are the lack of sub-pixel resolution without interpolating the reference frames, and the inability to estimate vectors at motion boundaries and areas of little texture. This thesis extends the method from simple minimization of SAD with the introduction of the SAD-map; the distribution of SAD values over a search range of offset is considered instead. Using this distribution, sub-pixel motion vector can be evaluated by models without actually interpolating reference frames. This interpolation-free sub-pixel refinement achieves within 75% of the performance achievable by actual interpolating the reference frame. A novel reliability measure and a smoothness constraint are proposed and applied in a novel queue-based block matching algorithm (QBMA). QBMA produces a field both lower in entropy for better compression, and a more natural field better suited for global motion estimation (GME) and motion segmentation.

To further improve the coding efficiency in motion estimation, another concept introduced is the global motion estimation which (i) provides a more compact representation of the motion field; and (ii) allows the reference frame to be warped, which represents a better match to the input frame than the original reference frame. The former property reduces the motion vector field information, thus reducing the number of bits required to code the motion vectors. The latter reduces the displaced frame difference energy, thus lessening the number of bits required to code the residue at any specified quality. Two novel methods are introduced which are suitable for real-time applications, one based on iterative regression (SAD-based iterative regression GME, SIRGME) and the other based on the Hough Transform (HT) called the progressive Hough-transform GME (PHGME). The SIRGME algorithm can be readily adopted in DSP-based algorithm for real-time applications; with the motion vector field obtained by QBMA and the use reliability based on the distribution of SAD-map, the global motion obtained by SIRGME offers a more accurate initial estimate. This improves the chances of reaching the global minimum, and the rate of convergence of SIRGME. With the much superior robustness towards outliers, using Hough Transform for GME provides a much more accurate estimate of the motion vector field. The high complexity in HT is alleviated greatly in the proposed PHGME via various progressive optimization techniques, bringing it a step closer to real-time implementation.

Although global motion estimation offers great improvements in motion estimation, especially in scenes dominated by a single global motion, scenes exist where more than one motion dominates. The concept of global motion is further extended to motion segmentation, which is viewed as partitioning a picture into several regions, each with its own global motion parameters. Global motion estimation becomes a special case of motion segmentation with a single segment. The main problem of motion segmentation is complexity and robustness. The Expectation-Maximization (EM) algorithm augmented by various simplification methods (Adaptive EM segmentation, AEMS) is introduced into the motion segmentation framework for reduced complexity. The Hough-transform is further incorporated into the EM algorithm as an initialization step to introduce robustness; and a simplifying segment merging algorithm is used to reduce the bitrate for segmentation information, together with the pre- and post-processing steps. The resulting proposed method, Pre/Post- AEMS (PPAEMS), offers a further reduction in bit rate at reduced complexity and improved robustness.

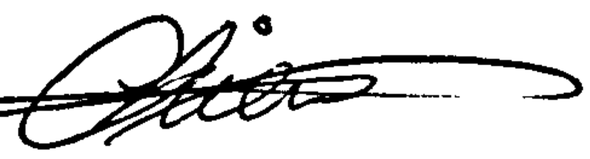
The algorithms proposed throughout this thesis are based on the SAD-map and its accompanying concepts; these algorithms are computationally tractable and are highly parallel in nature; a sub-pixel modelling algorithm is also proposed which circumvents the resolution problem of block-matching without frame interpolation. The accompanying improvements to the EM algorithm and HT are also low-complexity and entails reasonable memory requirements. Consequently, the algorithms proposed are all suitable for real-time applications, either with a DSP-based system or a hardware VHDL-based platform.

Acknowledgements

I am speechless, overwhelmed by the unbelievable fact that this thesis is finally completed. The excruciating journey has been made possible with my wife Puay Ling, to whom I owe the greatest gratitude for the understanding and thoughtfulness to see me through all the good and bad times. The new addition to the family, my daughter Vera, has given me immense motivation towards, as well as pleasant distractions away from, the completion of this thesis. Special thanks are extended to my supervisors Prof. Dave Bull and Prof. Nishan Canagarajah, for their invaluable advice and uncompromising guidance towards a successful end of my PhD endeavour. I am also grateful to my peers in Merchant Venturers' Building, James Chung-How, Przemyslaw Jan Czerepiński, Robert O' Callaghan, Andreas Constantinou, Mohammed Al-Mualla, and especially Paul Hill for proof-reading my otherwise unintelligible thesis.

Author's Declaration

I declare that the work presented in this thesis was carried out in accordance with the Regulations of the University of Bristol. The work is original except where indicated by a special reference in the text. No part of the dissertation has been presented to any other University for any degree either in the United Kingdom or overseas. Any views expressed in the dissertation are those of the author and in no way represent those of the University of Bristol.

SIGNED ~~D~~ 

DATE: 28/03/2005.

Contents

List of Figures

List of Tables

List of Notations

Digital image representation

x	Horizontal component of image co-ordinate
y	Vertical component of image co-ordinate
$\mathbf{p} = \begin{bmatrix} x \\ y \end{bmatrix}$	Vector representation of image co-ordinate (x, y)
$I(x, y) = I(\mathbf{p})$	Image intensity at co-ordinate $\mathbf{p} = (x, y)$
W	Image width
H	Image Height
B	Number of bits representing pixel intensity. A value of 8 indicates a pixel value of [0…255]
$\Lambda = \{\mathbf{p} \in [0, W-1) \times [0, H-1)\}$	Sampling lattice of an image
$\mathbf{I} = \{I(\mathbf{p}) : \mathbf{p} \in \Lambda\}$	Image intensity field.

Digital video (image sequence) representation

t	Temporal reference or time index of current video image
T	Total duration or number of images in a video sequence.
$I(x, y, t) = I(\mathbf{p}, t) = I_t(\mathbf{p})$	Intensity at co-ordinate $\mathbf{p} = (x, y)$ of an image at time t
$\mathbf{I}_t = \{I_t(\mathbf{p}) : \mathbf{p} \in \Lambda\}$	Intensity field of image at time t

Motion-related representation

u	Horizontal component of a vector
v	Vertical component of a vector
$\mathbf{v} = \begin{bmatrix} u \\ v \end{bmatrix}$	Vector representation of a motion vector
$\mathbf{v}_t(\mathbf{p}), \mathbf{v}(\mathbf{p})$	Motion vector at location $\mathbf{p}$ during time t
$e_t(\mathbf{p}) = I_t(\mathbf{p}) - I_{t-1}(\mathbf{p} - \mathbf{v}(\mathbf{p}))$	Displaced frame difference at location $\mathbf{p}$ due to motion vector $\mathbf{v}(\mathbf{p})$

$\theta = \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{k-1} \end{bmatrix}$	Global motion vector of a motion model with k parameters
$\mathbf{v}_t(\mathbf{p};\theta)$	Motion vector at location $\mathbf{p}$ during time t due to global motion with parameter vector,θ.
$e_t(\mathbf{p};\theta) = I_t(\mathbf{p}) - I_{t-1}(\mathbf{p} - \mathbf{v}(\mathbf{p};\theta))$	Displaced frame difference at location $\mathbf{p}$ due to global motion with parameter vector, θ.
$I_{t-1}(\mathbf{p};\theta)$	Intensity of $(t\text{-}1)^{th}$ image at location due to global motion with parameter vector, θ.

List of Abbreviations

Abbreviation	Description	Page
AEMS	Adaptive EM-based segmentation	170
BMA	Block Matching Algorithm	32
CVS	Candidate Vector Adaptation	169
EM	Expectation-Maximization	161
GME	Global Motion Estimation	32
HT	Hough Transform	131
IRGME	Iterative Regression for GME	108
ISE	Insignificant Segment Elimination	170
LME	Local Motion Estimation	32
MCS	Motion Candidacy Spread	65
MotSeg	Motion Segmentation	151
PPAEMS	AEMS with Pre- and Post-processing	170
QBMA	Queue-based BMA	70
QSS	Queue-based Segmentation Simplification	171
RGME	Regression-based GME	121
SAD	Sum-of-absolute-difference	35
SAD-map	Distribution of SAD as a function of motion vector	64
SIRGME	SAD-map-based IRGME	108
SPHMS	Hough Transform-based Motion Segmentation	170

Chapter 1:

Introduction

1.1 Background

As communication bandwidth and storage technologies advanced into the twenty-first century, numerous real-time applications, deemed too complex in the past, have become common-place. Amongst these applications, digital video is the undisputed leader in terms of the pace at which it has been assimilated into everyday use and the breath and extent of its influence. Video surveillance, Digital TV, DVDs and video-phones are but a few examples of its numerous applications.

The advantages of using digital over analog video are in large extent similar to that of all other media. Digital signals are more robust to transmission noise and storage of digital data allows perfect reconstruction; that is, no degradation in quality when digital video is copied. Various digital media can be meshed together to provide a richer user experience for the entertainment and education industry. Encryption technology also allows security and anonymity in the media which is not possible in its analog counterpart. Perhaps, most important of all is the possibility of processing digital video in increasing variety of ways. Error-resilience and video compression are the two most prevalent processing. Digital video compression is the main focus of this thesis.

Digital video compression owes its success much to the fact that it exhibits high redundancy within individual frames as well as high similarity between neighbouring frames. These redundancies, when effectively exploited, can lead to a substantial reduction in the bit-budget for representing the video sequence. To ensure interoperability and minimize time to market, various video coding standards have been proposed. Examples include the ITU's H.263, used in video-communications applications, and MPEG-2 used in DVDs and digital TV.

Whereas intra-frame redundancies are mainly reduced by transform coding, the inter-frame redundancies can be effectively removed by motion estimation. Block matching algorithms are used extensively for motion estimation in all video compression standards due to their low complexity. However, many pixel-based algorithms exist, which are more robust to noise and produce more accurate motion field. These methods, like that proposed by Horn and Schunck [Hor-81], are computationally intractable and cannot be readily adopted by digital signal processing and hardware designers. This thesis tries to bridge this implementation gap by efficiently integrating some of the

more complex algorithms and concepts into the simple block matching framework. This would improve the capability of existing video coding systems to estimate motion fields more efficiently and robustly.

In addition, more complex methods of motion estimation, such as global motion estimation and motion segmentation, can provide higher compression and should be adopted and used in the next generation of video coding standards. Existing algorithms related to these methods are too complex to be implemented real-time. Two particular algorithms, the Hough Transform and the Expectation-Minimization estimation are investigated, simplified and embodied into the global motion estimation and motion segmentation process.

1.2 Research Objectives

Motion-related processing in video coding systems usually involves a direct block matching algorithm (BMA), which is both computationally tractable and effective. The main drawbacks are the lack of floating-point sub-pixel resolution and artificial discontinuities due to the aperture problem. One of the research objectives of this thesis is to find means of mitigating these problems by adopting existing, more complex methods and reducing their complexities to make them realizable in real-time coding systems. The proposed framework makes use of the SAD (sum-of-absolute-difference) distribution (termed the SAD-map) to derive a few new concepts, leading to the Queue-based BMA (QBMA).

The motion vector field representation constitutes a large proportion of the bit budget in state-of-the art video coding systems and in low bit-rate applications where the residual texture information is highly quantized. The second aim of this thesis' work entails finding more compact representations of this field. As a result, novel methods of global motion estimation (GME) and motion segmentation (MotSeg) are used on the QBMA vector field. In particular the Hough Transform (HT) and expectation-minimization (EM) are utilized in this thesis to reduce complexity and improve robustness of GME and MotSeg.

The main thrust of this thesis is to attempt to tackle the decade-old problem of BMA-based motion estimation from a different angle, based on more robust and sophisticated algorithms. The main targets include:

- Reduce complexity and processing time.
- Improve the quality of the motion vector field by the block matching algorithm (BMA)
- Provide a more compact representation of the motion vector field through global motion estimation (GME) and motion segmentation (MotSeg).
- Provide an improved warped version of the reference frame which better matches the input frame for multiple-pass BMA.

The main aim is to provide the intra-coder (texture coding) with the displaced frame difference (DFD) (or textural residual) containing the little entropy as possible. The principal concern in this thesis is

lossless inter-frame processing. It is believed that providing a lowest entropy residual to the texture coder and quantizer is crucial to having a good rate-distortion curve as the amount of information to start with is minimized. Another part of the thesis is concerned with removing redundancy in the motion vector field without affecting the textural entropy. This is not related to the rate-distortion optimization problem, but is aimed at reducing the overhead of coding the motion information which constitutes a higher proportion of the bit budget in recent coding standards where block sizes get smaller and target bit-rates get lower.

1.3 Thesis Structure

The thesis documents the result of three main areas of author's research work. Chapter 2 lays down the basics of digital video and introduces the necessary nomenclatures required in subsequent chapters. Chapter 3 and 4 feature the local motion estimation, in particular the block matching algorithm (BMA).

Chapter 3 provides some historical and theoretical backgrounds on motion estimation; chapter 4 introduces the novel approach to BMA which facilitates the introduction of smoothness constraint on the motion vector field, without degrading the predictive capability of motion estimation. The SAD-map is introduced which forms the basis of the Queue-based BMA, QBMA. In addition, a novel reliability measure is introduced which out-performs other measures in terms of registering how well a block's motion vector is estimated. The chapter also introduces a novel means of circumventing the lack of motion vector resolution in BMA without frame interpolation. The combined QBMA and sub-pixel modelling will be used in algorithms proposed in subsequent chapters.

In chapters 5 and 6, motion estimation is given a global prospective. Chapter 5 begins by introducing the advantages of global motion estimation and reviewing some of the solution common to this problem. The traditional regression-based global motion estimation (GME) method is improved with the use of SAD-map. Termed the SAD-map-based iterative regression GME (SIRGME), this novel method provides a more compact motion vector field than the traditional GME (TGME). Chapter 6 introduces another approach to GME, using the Hough Transform (HT). The thesis introduces the PHGME which introduces robustness to SIRGME without imposing excessive processor and memory requirements to the video coder.

The theory used in global motion estimation is extended into motion segmentation in chapter 7. The expectation-maximization (EM) algorithm is introduced and adopted in motion segmentation. The SAD-map is again used as the input to the EM-based motion segmentation. The Hough transform is also used to provide an initial estimate of the segmentation and the parameters. The combination of SAD-map, HT and EM is shown to provide a low-complexity solution to the usually computationally intractable problem.

All algorithms described in chapters 4 to 7 are applied in simulations performed on an 850MHz Pentium 3 Laptop with 512 Mbytes of RAM. The thesis ends with Chapter 8, which provides some useful conclusions drawn from the simulations and observations on the algorithms described in the preceding chapters. This final chapter also gives some recommendations for future work.

Chapter 2:

Basics of Digital Video

This chapter lays down the necessary ground work for succeeding chapters. First the basic nomenclature of digital video is used in this thesis. Terms and representation required for the following chapters are defined, followed by the common formats and parameters pertaining to digital video. Next, the statistical depictions of digital video are discussed and how redundancies can be exploited is investigated. For completeness, the common video compression standards, namely H.261, H.263, H.264, MPEG-1, 2 and 4, are briefly introduced.

2.1 Anatomy of Digital Video

2.1.1 Digital Video Representation

A digital image is both discrete and quantized. It is a discrete spatial distribution of intensity $I(x,y)$ where $x \in \{0,1,\ldots,W-1\}$ and $y \in \{0,1,\ldots,H-1\}$, W and H being the number of horizontal pixels and vertical lines in the image respectively. Each pixel can only take values from a quantized set of values. The extent with which these values can take depends on the number of bits, B, used in the imaging system. The common values are $B=8$ ($I \in \{0,1,\ldots,255\}$) or $B=10$ ($I \in \{0,1,\ldots,1023\}$). A digital video sequence as illustrated in Figure 2.1 is a sequence of digital images whose pixel values vary with time, t. It can be represented as $I(x,y,t)$ where $t \in \{0,1,\ldots,T-1\}$, T being the number of images in the sequence, or a contiguous part of a long sequence of interest. In real time applications T may be infinite. Hence a grey-scale digital image is a mapping single-valued mapping of 3 independent variables:

$$I:[0,W-1]\times[0,H-1]\times[0,T-1]\rightarrow[0,2^B-1] \qquad \text{Eq 2-1}$$

By using the position vector representation $\mathbf{p}=(x,y)$ and putting the emphasis of t at the frame level instead of the pixel level, the following representations of an image sequence are interchangeable:

$$I(x,y,t)=I(\mathbf{p},t)=I_t(\mathbf{p}) \qquad \text{Eq 2-2}$$

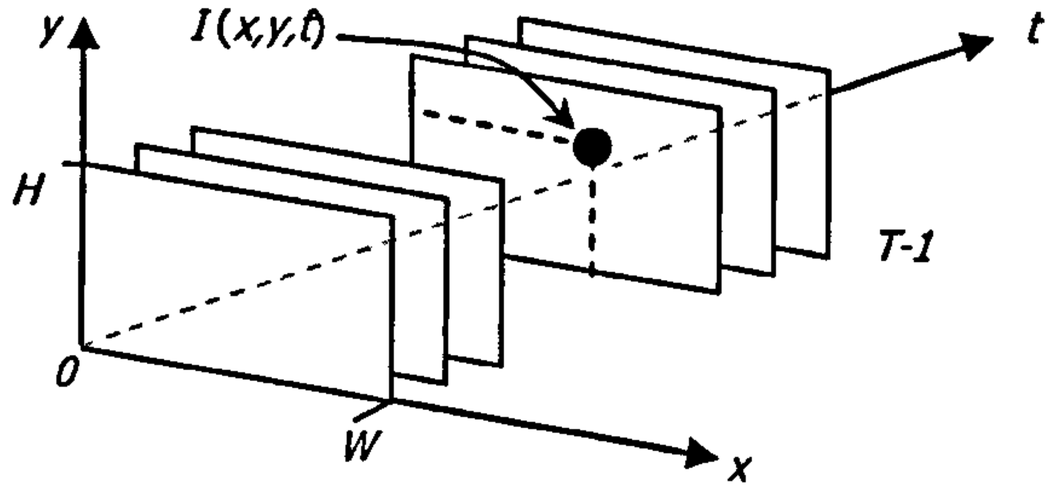


Figure 2.1 Graphical depiction of a video sequence $I(x, y, t)$.

2.1.2 Colour Representation

Colour video sequences are collections of 3 mappings (refer to Eq 2-1) which depend on the colour system used. The common colour component systems are the RGB and the YC_bC_r; others include the YIQ, YUV and HSI. The RGB system represents the colour images in the intensities of the 3 primary colours red, green and blue.

$$R:[0,W-1]\times[0,H-1]\times[0,T-1]\rightarrow[0,2^B-1]$$
$$G:[0,W-1]\times[0,H-1]\times[0,T-1]\rightarrow[0,2^B-1] \qquad \text{Eq 2-3}$$
$$B:[0,W-1]\times[0,H-1]\times[0,T-1]\rightarrow[0,2^B-1]$$

However, the digital video industry frequently uses the YC_bC_r system that represents colour images as one luminance component Y and two chrominance (colour difference) components, C_b and C_r. The conversion between these two colour systems can be represented as the following matrix equation:

$$\begin{bmatrix} Y \\ C_b - 2^{B-1} \\ C_r - 2^{B-1} \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.147 & -0.289 & 0.436 \\ 0.615 & -0.515 & -0.100 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \qquad \text{Eq 2-4}$$

The natural colour differences varies as signed integers between $[-2^{B-1}, 2^{B-1}-1]$, the 2^{B-1} offsets in Eq 2-4 causes C_b and C_r to take up unsigned values in the range $[0, 2^B-1]$. Representing video in luminance and chrominance components can be traced back to the analog video broadcast industries, where chrominance information is added to the black and white television signals for compatibility. The human visual system is more sensitive to the luminance variation than the chrominance variation, so the C_b and C_r components can be sub-sampled without apparent loss in the subjective video quality. A few chrominance sub-sampling spatial schemes are commonly used, these include the 4:1:1 system,

4:2:2 system and the 4:2:0 system. Figure 2.2 illustrates the relative distribution of the luminance and the chrominance pixels:

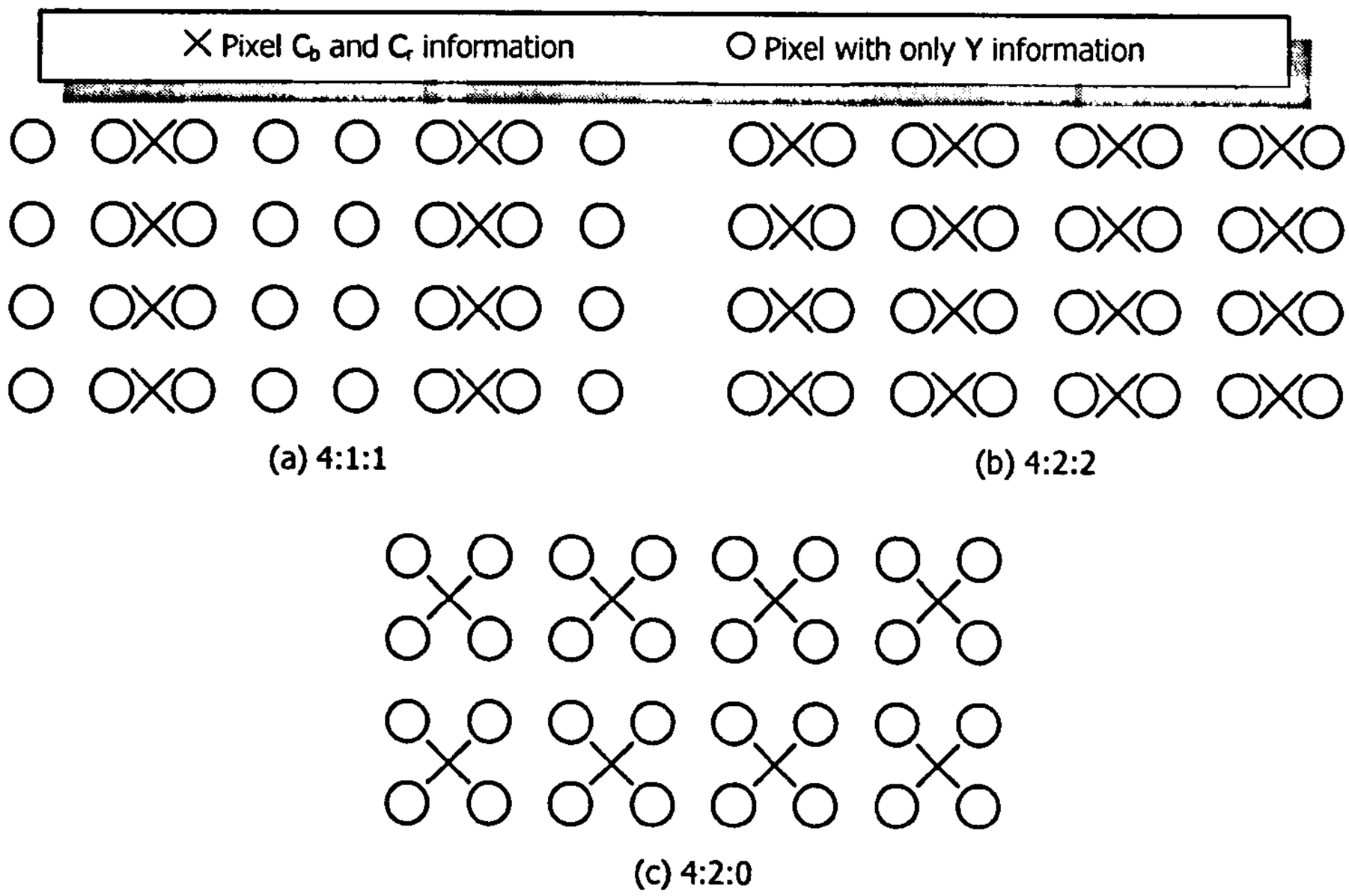


Figure 2.2 Distribution of Y, C_b and C_r pixels in various colour sub-sampling formats.

2.1.3 Common Video Formats and Applications

The *International Consultative Committee for Radio* Recommendation 601 (CCIR-601) defines digital video format for the exchange and storage of digital formats. As a legacy of analog video, two size formats and refresh rates are used, the 525/60 of NTSC system uses 720x480 pixels at 30 frames per seconds (fps) and the 625/50 or PAL system uses 720x576 pixels at 25 frames per second. Pixel rates are both set at 13.5 MHz; a colour space of YC_bC_r at 4:2:2 is used. All pixel data are quantized to 8 bit-wide code. The Y component has its value clipped between 16 and 235, with 16 representing total darkness and 235 representing full-scale brightness. As C_r and C_b components are difference values, they are zero-offset with the value 128 and the clipped between 16 and 240 with a value of 128 representing zero colour difference. Values not in the range are used for synchronisation purposes.

To cater for applications with lower video quality, CCIR-601 recommends the SIF format (352x288@25fps, 352x240@30fps) and the QSIF format (176x144@25fps, 176x120@30fps), both with 4:2:0 chrominance sub-sampling.

To accommodate transferring sequences between the CCIR-601 formats, a family of *Common Intermediate Formats (*CIF) is proposed and widely used in the video processing community. The format adopts the 625/50 resolutions as illustrated in Table 2.1 and the 525/60 refresh rate of 30 fps. Like SIF, CIF uses the 4:2:0 YC_bC_r colour space.

Table 2.1 Resolution of various members of the CIF family.

	Luminance		**Chrominance**	
	Pixels per line	Lines per frame	Pixels per line	Lines per frame
SQCIF	128	96	64	48
QCIF	176	144	88	72
CIF	352	288	176	144
4CIF	704	576	352	288
16CIF	1408	1152	704	576

In this thesis, three typical types of video sequences are considered. The first type is used in low-complexity mobile video communication applications where the user is less demanding on the quality but is more sensitive to the delays. The target bit rate of such system would typically be below 100 kbps. The typical resolution of such applications would be that of the QCIF format. To reduce processing times and bit-rates, a frame rate of 10 fps is used (based on skipping 2 frames out of the assumed 30 fps input sequence). The second application is for transmission of video via Wireless LAN in home, office or public premises. The quality requirements are higher and off-line processing may be possible; the typical bit-rate of this application ranges from 500 kbps to 24 Mbps. The video format would likely be CIF@30 fps. The third application is video surveillance in personal home and public places. The requirement of this application is more scalable, depending on whether the video stream needs to be transmitted or stored; however, for obvious reasons, video compression has to be done real-time. In these two cases the QCIF@10fps and CIF@30 fps are also reasonable formats. In summary, simulations in subsequent chapters will be based on the two stated sizes and frame rates typical to the three applications mentioned above.

2.2 Characteristics and Quantitative Measures of Digital Video

2.2.1 Statistical Characteristics of Digital Video

The two most important forms of redundancy in image signals are statistical redundancy and subjective redundancy [Has-98]. We represent the amount of redundancy as the reduction in entropy in the video data. We first define various entropies and illustrate how statistical redundancies exploited in video coding. The amount of information carried by an image can be represented by its entropy [Sha-48] as defined as:

$$H(I) = p_i \log_2 p_i \qquad \text{Eq 2-5}$$
$$p_i = P(I(x,y,t) = i) \quad i \in [0 \ldots 2^B)$$

If an image data is fully random, its entropy will be close to B bits; the lower the entropy value, the less random the data. Correlation amongst pixel values can be indicated by the difference equations along its x- and y- axes:

$$I_x(x,y,t) = I(x,y,t) - I(x-1,y,t) \qquad \text{Eq 2-6}$$
$$I_y(x,y,t) = I(x,y,t) - I(x,y-1,t)$$

On the other hand, the entropy of the derivative with respect to time defined below, gives an indication of how much the images change in time. It is commonly known as the frame difference (FD).

$$\dot{I}(x,y,t) = I(x,y,t) - I(x,y,t-1) \qquad \text{Eq 2-7}$$

Another form of temporal entropy deals with moving objects. Pixels belonging to a stationary object will not change pixel intensities between two frames. Pixels belonging to a moving object change their pixel location from frame to another. The displaced-frame difference (DFD) of a pixel at (x, y) is defined as the difference between the pixel value of the current frame and a corresponding pixel at location $(x+u, y-v)$ of a previous frame.

$$DFD(x,y,t;u,v) = I(x,y,t) - I(x+u, y+v, t-1) \qquad \text{Eq 2-8}$$

The vector $\mathbf{v} = (u,v)$ is the displacement of pixel (x, y) due to object motion. The set of motion vectors (u,v) within a picture is collectively known as the motion vector field and process of finding (u,v) is commonly known as motion estimation.

In order to reconstruct a picture with motion estimation, the DFD information alone is insufficient; information about the motion (u,v) is also required. Hence the last entropy to be defined, called the motion vector entropy, is the amount of information carried by the motion vector field. A picture with a uniform motion (say panning by a camera) would contain very low entropy than a picture with many independently objects moving in different directions.

The entropies of three QCIF sequences are examined: AKIYO, FOREMAN and STEFAN. The plots are presented in Figure 2.3, Figure 2.4 and Figure 2.5, with the three pictures depicting the first, middle and last frames of each sequence. The entropies and the key used are listed below:

- H(I) – pixel entropy, $H(I)$
- H(Ix+Iy) – spatial derivative entropy, $H(I_x)+H(I_y)$
- H(I') – frame difference entropy, $H(i)$
- H(DFD+MV) – combined entropy of displaced frame difference and motion vector field, $H(DFD)+H(\mathbf{v})$

As indicated in the figures, pixel entropies of both sequences are around 7 bits, which means a possible reduction of 12.5% can be achieved by simply losslessly compressing the raw image sequence. The entropies of the spatial partial derivatives of all three sequences are lower (4.2, 5.0 and 6.0 bits respectively), indicating a strong correlation amongst neighbouring pixels, this redundancy can be removed by predictive coding within the frame, or commonly known as Intra-Coding.

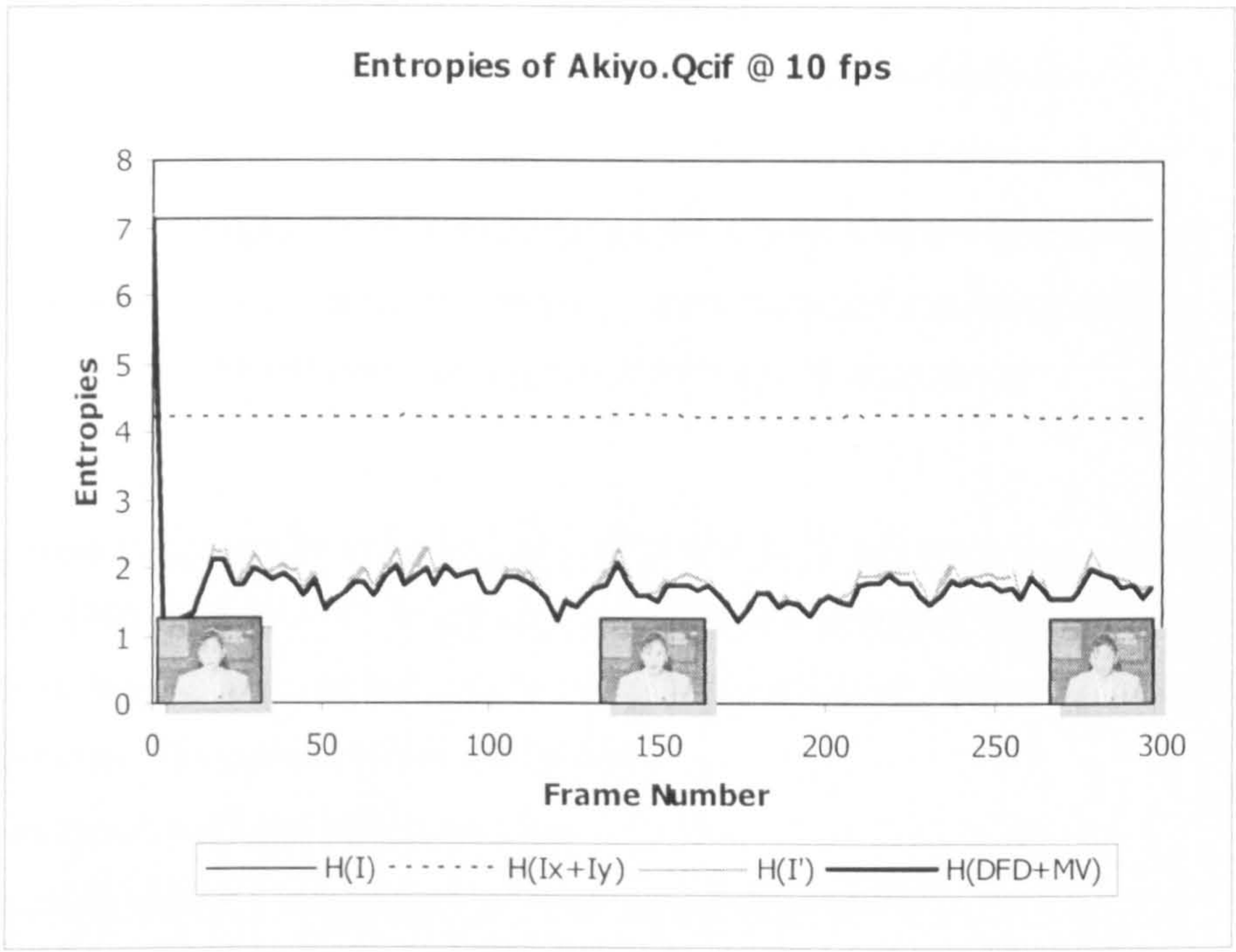


Figure 2.3 Entropies of AKIYO.QCIF sequence.

The entropies of the FD and DFD+MV vary interestingly amongst the three sequences. AKIYO.QCIF sequence is relatively static and hence simple frame differencing can remove a lot of inter-frame redundancies; motion estimation does reduce the entropies further, but not by much. Both FD and DFD reduce the temporal redundancies of the sequence; prediction of this sort is termed as Inter-Coding. Both FOREMAN.QCIF and STEFAN sequences have a fast moving objects and simple frame differencing does not necessarily outperform Intra-Coding schemes. Motion estimation reduces the entropy by a substantial amount.

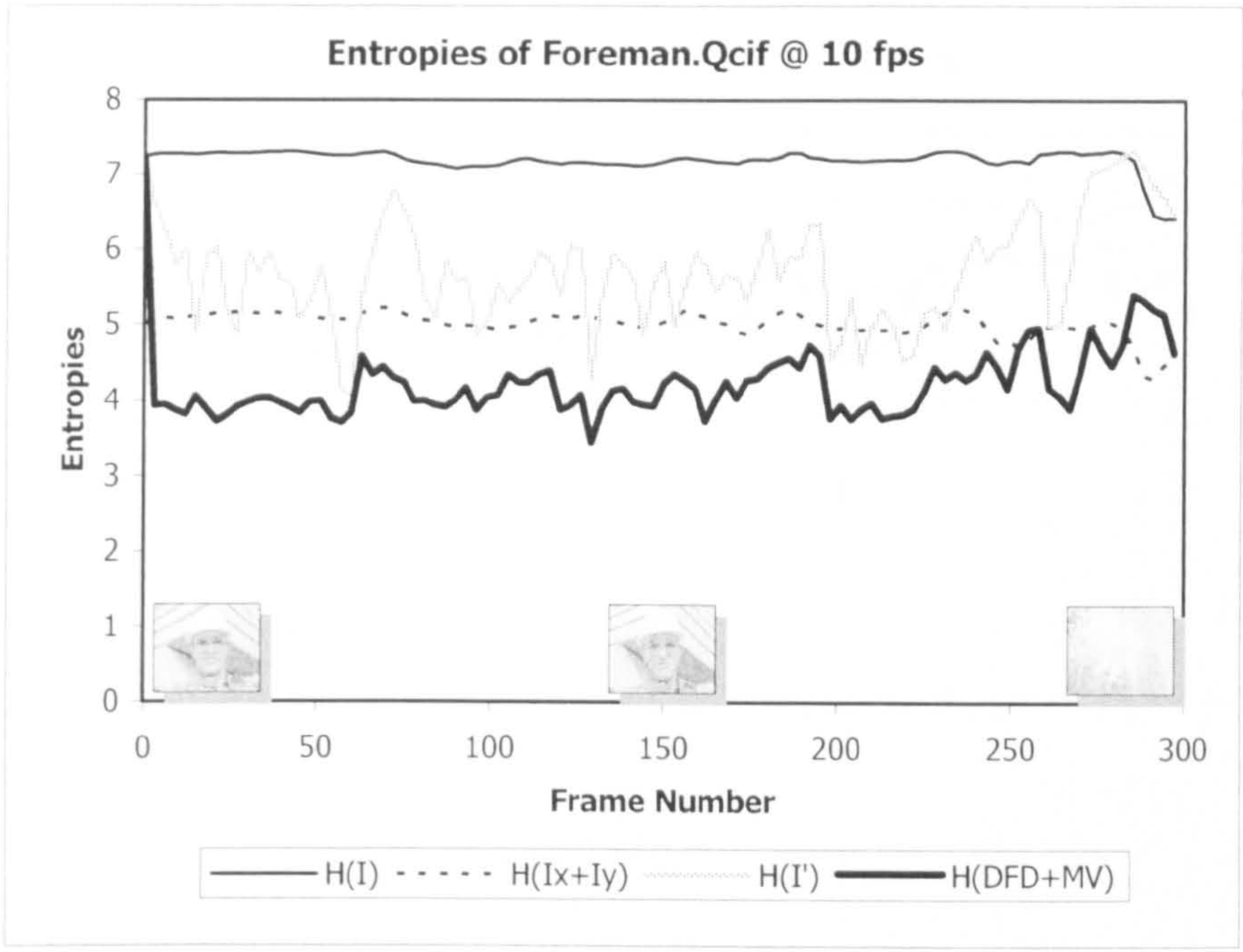


Figure 2.4 Entropies of FOREMAN.QCIF sequence. H(I) = pixel entropy, H(Ix+Iy)= partial derivative entropy; H(I')=temporal derivative entropy; H(DFD+MV)=entropies of DFD and displacements.

An interesting observation can be made with the STEFAN.QCIF sequence – in certain frames (around frames 100 and 190) H(DFD+MV) is higher than H(Ix+Iy), implying that motion estimation is no better than simple Intra-Coding. In fact, this is the case in parts of all frames and is more dominant in the mentioned frames. The motion failure can be due to:

- The motion is too large to be estimated.
- The motion does not exist due to previously occluded regions
- The motion is too complex to be estimation accurately enough, e.g. objects undergoing expansion and distortion
- Object not present in the previous scenes appearing in current frame.

Regardless of the reason, in order to achieve good compression efficiency, any video compression systems should have the mechanism to change between Inter- and Intra-Coding adaptively in different regions of a frame.

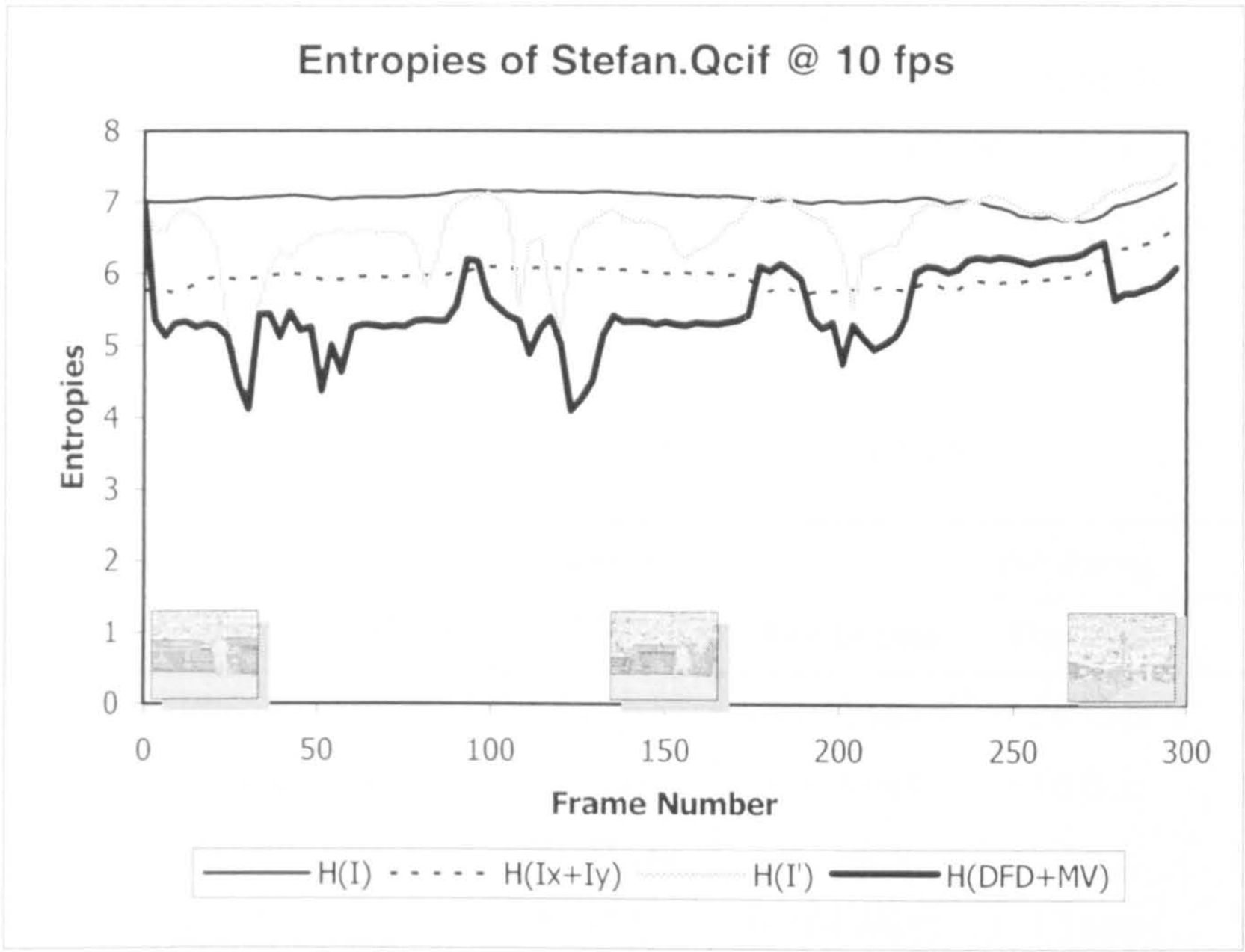


Figure 2.5 Entropies of STEFAN.QCIF sequence. H(I) = pixel entropy, H(Ix+Iy)= partial derivative entropy; H(I')=temporal derivative entropy; H(DFD+MV)=entropies of DFD and displacements.

The entropy of the temporal derivative, on the other hand, is very different in AKIYO and STEFAN sequences. The AKIYO sequence demonstrates a much lower H(I'), as evidenced by the relatively static nature of the sequence. STEFAN, a typical sports sequence, exhibits a much more dynamic inter-frame activity. Both sequences, however, can still be compressed further by exploiting this inter-frame redundancy.

2.2.2 Video Bit Rates and Compression Ratio

The amount of data is measured in bits, which is the number of binary symbols required to represent the data. The following bit rates are commonly used to represent video data:

- Bits per frame (bpf)
- Bits per pixel (bpp)
- Bits per second (bps)

The essence of all compression is throwing data away. If the data to be discarded are purely redundant and is required for complete construction, the compression is termed lossless compression; if quality is reduced as a result of the compression, the process is known as lossy compression. The effectiveness of a compression scheme is indicated by its "compression ratio," which is determined by dividing the

amount of data to begin with by the amount of data after compression. Through the removal of redundancies and sometimes at the expense of fidelity, a compression system reduces the entropy of the video data, thus reducing the bit-rates required to store or transmit the bit stream. To have some idea of the compression ratio required in common application, we refer to Table 2.2 (raw bit rates of some common video formats) and Table 2.3 (typical target bit-rate required by current communications and storage system).

Table 2.2 Raw bit rates of popular formats.

Format Name	Size Format	Colour format	Frame Rate	Bit Rates		
				Per frame	Per pixel	Per sec.
HDTV	1280x720	4:2:2	60 fps	18.432 Mbpf	20 bpp	1.1 Gbps
CCIR601(PAL)	720x576	4:2:2	25 fps	6.6 Mbpf	16 bpp	166 Mbps
CIF	352x288	4:2:0	29.97 fps	1.2 Mbpf	12 bpp	36.5 Mbps
QCIF	176x144	4:2:0	29.97 fps	0.304 Mbpf	12 bpp	9.1 Mbps

For a HDTV system requiring a 20 Mbps to transmit its 600 Mpbs raw-video content, the compression system needs a compression ratio of 30:1. In a video-phone application, a typical video requires a QCIF format at 10 frames per second (fps), which results in a raw bitrate of 3 Mbps; at a channel capacity of 24 kbps, the encoder needs to be compressing at a rate of 125:1.

Table 2.3 Bit rates of popular applications.

Application	Bit rate
POTS Videophone	10-25 kbps
ISDN Video Conferencing	384 kbps
VideoCD	1.5 Mbps
DVD	2-10 Mbps
WLAN video	0.1-10 Mbps
HDTV	20 Mbps

2.2.3 Reconstruction Fidelity

If data is compressed without any loss, perfect reproduction is possible; however, the compression process for video data is usually lossy in nature. As compression ratio increases, reconstructed data bear less resemblance to the original. An objective measure commonly used to represent the amount of

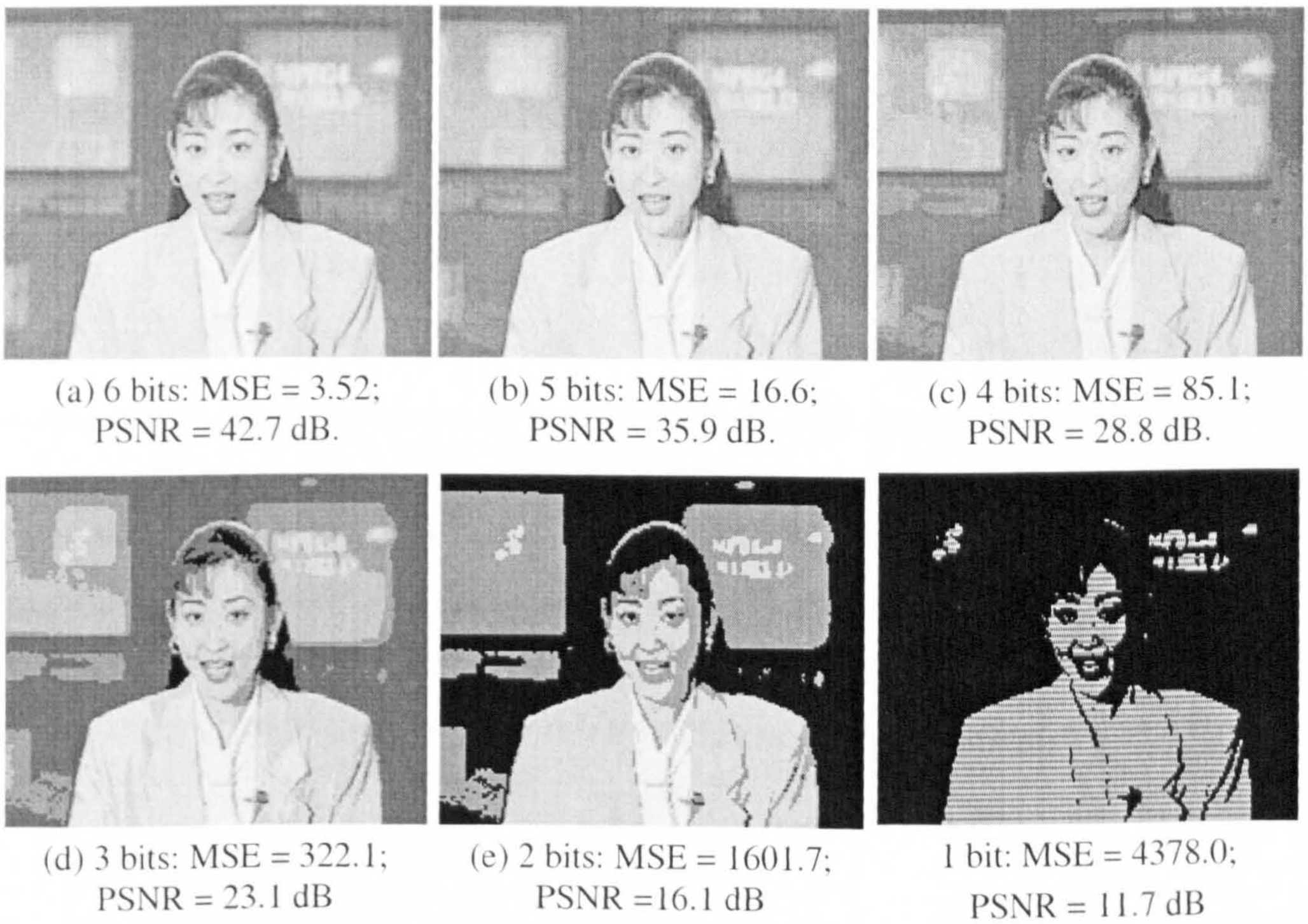
degradation is the mean square error (MSE) between the reproduced and original image (I' and I respectively):

$$MSE(I,I') = \frac{1}{W \times H} \sum_{y=0}^{H-1} \sum_{x=0}^{W-1} \left(I(x,y,t) - I'(x,y,t) \right)^2 \qquad \text{Eq 2-9}$$

Another measure of fidelity is the peak signal to noise ration (PSNR):

$$PSNR(I,I') = 10 \log_{10} \left(\frac{(2^B - 1)^2}{MSE(I,I')} \right) \qquad \text{Eq 2-10}$$

As PSNR provides a positive measure to fidelity and its logarithmic scale provides a more consistent indication of perceived picture quality, it is very widely used indeed. In Figure 2.6, the number of bits to represent an image of the QCIF AKIYO sequence is progressively reduced. The effect is similar to removal of least significant bits by quantization. A PSNR value in the range of 30 to 50 dB is present little distortion to the original image; any value below 25 dB becomes quite intolerable.



(a) 6 bits: MSE = 3.52; PSNR = 42.7 dB.

(b) 5 bits: MSE = 16.6; PSNR = 35.9 dB.

(c) 4 bits: MSE = 85.1; PSNR = 28.8 dB.

(d) 3 bits: MSE = 322.1; PSNR = 23.1 dB

(e) 2 bits: MSE = 1601.7; PSNR =16.1 dB

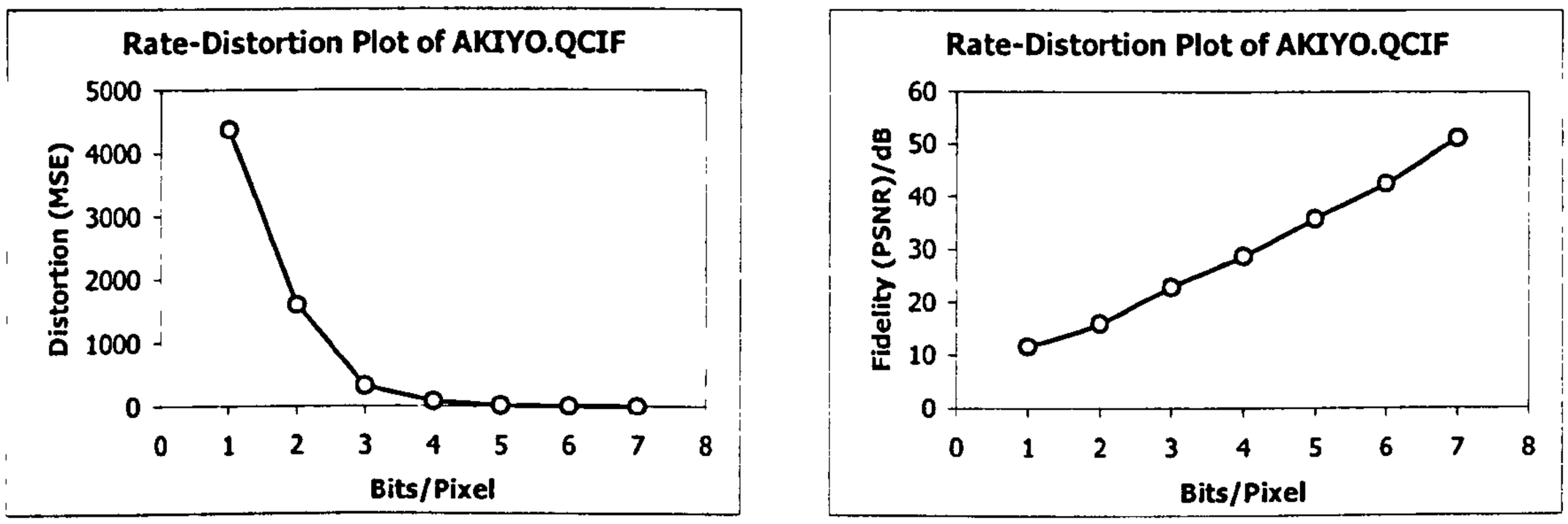
1 bit: MSE = 4378.0; PSNR = 11.7 dB

Figure 2.6 PSNR of images quantized at different bits

It should be noted that quantitative measures like MSE and PSNR provide a tangible measure of the amount of distortion brought about by the compression system; they do not take into consideration on how the viewers' response towards the distorted image. More subjective measures like the mean opinion scores are one of the recent attempts to incorporate subjectivity into the distortion measurements [Wan-02].

2.2.4 Rate-distortion Theory

The contribution of Shannon's theoretical analysis of the relationship between fidelity and coding rate has expedited the progress of research in video compression techniques. As Figure 2.7 illustrates, the PSNR increases with bit-rate while MSE decreases with bit-rate.



(a) (b)

Figure 2.7 Rate-Distortion plots of AKIYO.QCIF with quantization by bit truncation. (a) uses distortion in terms of MSE; (b) uses fidelity measure (PSNR).

A direct use of R-D plot is to estimate the performance of a coding scheme. Take the simple algorithm of bit truncation used in Figure 2.7 for example. In order to achieve a PSNR of 30 dB we need approximately 4 bits per pixel. Another use of R-D plot is for comparison performances of difference coding schemes. In Figure 2.8, we compare two schemes – (a) is bit-truncation as in Figure 2.7; and (b) the bit-truncation of frame difference. The R-D curve of (b) lies entirely above that of (a), indicating a (b) is a more superior coding scheme. At 4 bits/pixel (bpp), say (b), an improvement of 11 dB over (a) can be achieved.

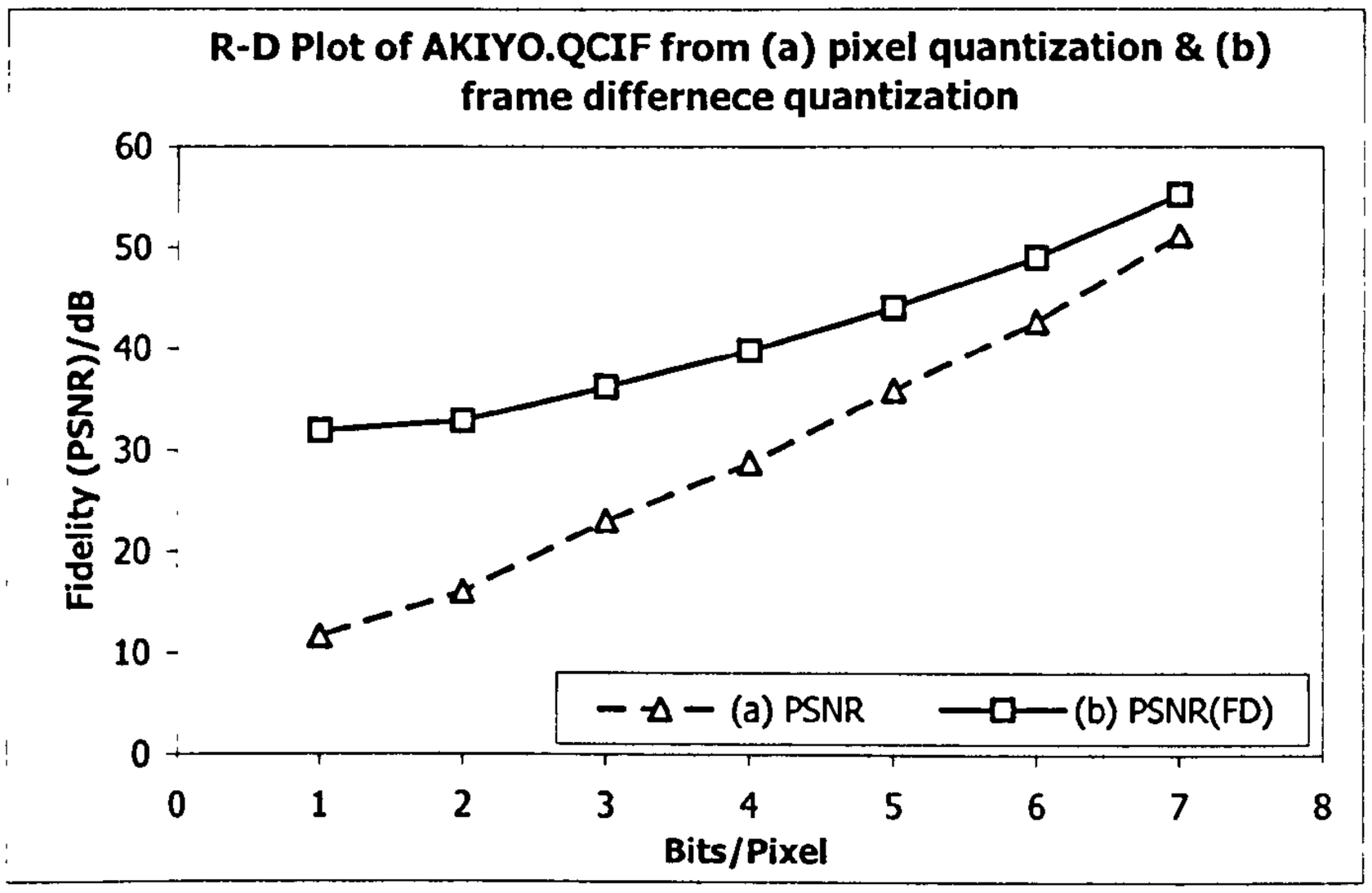


Figure 2.8 Rate-Distortion plots of AKIYO.QCIF of two simplistic coding schemes: (a) uses simple bit truncation of original image; (b) bit truncation of frame difference.

2.3 Video and Image Compression

Compression in general involves removing redundancies from the actual data. If the removal entails no loss in information from the original data, actual reconstruction can be carried out to recover the original data. This type of compression is termed lossless compression. Video data has abundance of redundancy to be exploited both spatially within a frame and temporally across frames. Whenever this redundancy is fully exploited but the amount of information is still too large to be either transmitted or stored, lossy compression is required to bring the bit-rate further down at the expanse of quality. As a result of lossy compression, the original video can only be recovered partially, thus reducing the quality of the video. How this quality is compromised with the bit-rate using different schemes is studied in the field of rate-distortion theory. This section describes the various classes of compression techniques used for video compression.

2.3.1 Entropy Coding: Lossless Compression

Since the beginning of the nineteen-eighties, the first-generation compression systems had based their techniques on lossless compression [Rei-97]. They are also called entropy coding, as the main target is to allocated longer code lengths to less probable data to lower the overall bit rate to the towards the

theoretical minimum, its entropy. Lossless processes have relatively low compression ratios, typically in the order of 2:1. Three most widely used entropy codes:

- Huffman coding
- Arithmetic coding
- Run-length coding

2.3.1.1 Huffman coding

The Huffman code (accredited to D. A. Huffman, 1952) is a prefix code which assigns codes of different lengths based on the a priori probabilities of the data.

A typical binary Huffman coding process is as follows:

1. Obtain a large enough of training data and form a histogram $p(i)$ where i is the pixel value and $p(i)$ is the probability of occurrence of i.
2. Arrange the probabilities in ascending order.
3. Merge the two codewords with the lowest probabilities to form a new aggregate code with the new probability as the sum of the probabilities of the two original codewords.
4. Repeat the merging process until all codewords are merged.
5. Build each codeword by splitting the aggregates in the reversed order, adding a one and a zero to each split.

The resulting code has codewords whose length is $\lceil p(i) \rceil$ where $\lceil \bullet \rceil$ denotes the integer ceiling. Hence Huffman coding is only optimal if the data probabilities are powers of 1/2, that is 1/4, 1/8 and so on. In other cases, an alternative coding scheme, the arithmetic coding, can provide a more optimum compression rate.

2.3.1.2 Arithmetic Coding

In contrast to the Huffman codes which work with integer numbers, Arithmetic codes make use of progressively small intervals of a floating point number to represent a sequence of symbols, thus resulting fractional bits representation.

As an illustration, take 4 symbols {a, b, c, d} with probabilities {0.1, 0.2, 0.3, 0.4}, the sequence "ddcab" can be coded as a floating point number within the range (0.93736, 0.93808).

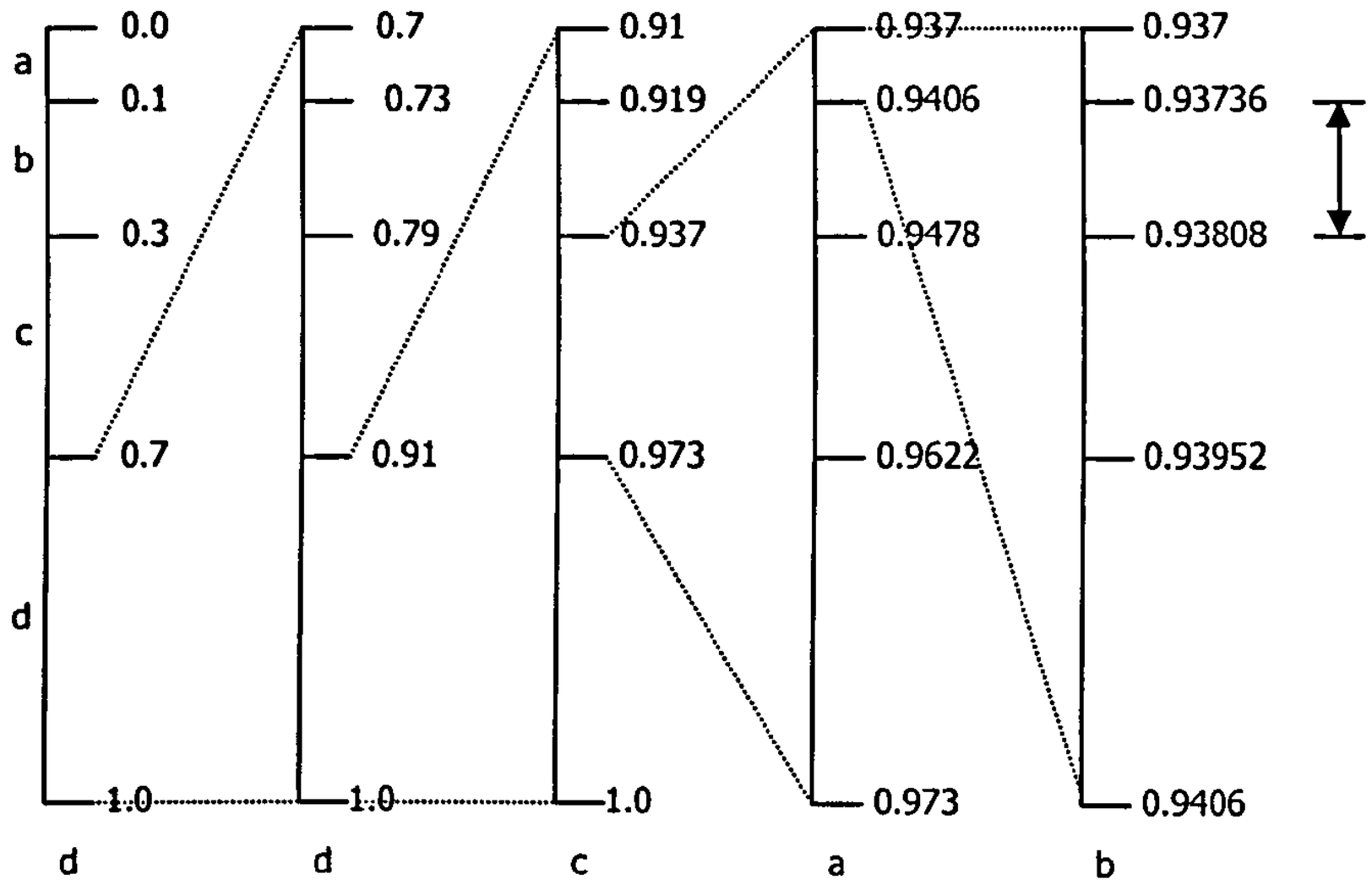


Figure 2.9 An illustration of arithmetic coding.

If we choose to represent the floating by 4 bits, we can code "ddcab" as {1111} ($\frac{1}{2}+\frac{1}{4}+\frac{1}{8}+\frac{1}{16}=0.9375$). Requiring just 4/5 = 0.8 bits per symbol.

2.3.1.3 Run-length Coding

For data sequences with a majority of zeros populated sparsely by non-zero values as depicted below, a useful compression scheme is run-length coding.

$$\{12,0,0,0,0,34,0,0,0,0,0,0,0,0,0,0,0,-20,10,0,0,0,0,0,0,0,0,0\} \qquad \text{Eq 2-11}$$

$$\Downarrow$$

$$\{(12,4),(34,11),(-20,0),(1,10)\}$$

The run-length code system represents the sequence as value-run pairs (v,r) where v denotes the value of the non-zero value and r denotes the number of running zeros following v. Assuming same number of bits are used to represent v and r as the original data, a compression ratio of 8/28=0.2857.

As lossless coding does not remove any video data in order to achieve compression, the decompressed image can be fully recovered. However such lossless compression alone is not enough for video data to be transmitted in most of the current communications channels. Lossy video compression systems use lossless techniques where they can, but the major part of bit savings come from discarding data. A video sequence to be processed is separated into two general groups of data. One group, containing all the important information, is transmitted losslessly; the other group, with all the less crucial

information, will have selected bits discarded. As the decompressor receives a truncated version of the original data, the reconstructed video can only bear a close resemblance to the original sequence. How much is the fidelity of the decompressed video depends largely on the nature of the input video sequence, the allocated bit budget and the efficiency of the compression algorithms. The following sections discuss the various means to remove redundancies of the video data to achieve much higher compression ratios.

2.3.2 Perceptual Coding

All second-generation techniques [Rei-97] make use of some properties of the human visual system (HVS) in the compression algorithms in order to achieve higher compression ratios while still maintaining acceptable image quality. Perceptual coding takes advantage of the non-uniformity between the perceptual quality of images and the amount of data required to represent them. Lossy compression systems take the characteristics of our eyes into account. A HVS is commonly modelled as a low-pass filter, a logarithmic nonlinearity, and a multi-channel signal-sharpening high-pass filter [Rei-97]. Four properties of the HVS are listed below:

- Non-linearity of intensity sensitivity – the sensitivity reduces as the background intensity increases.
- Non-separableability of temporal and spatial sensitivity – the threshold sensitivity depends on both the spatial and temporal frequencies. In fact, for regions with high spatial frequency, the HVS resembles a low-pass filter; for uniform regions, the response is essentially band-pass in nature.
- Directional anisotropy – the eye is more sensitive to horizontal and vertical frequencies than in the oblique directions, with minimum decreased by about 3 dB around 45°.
- Spatial and temporal masking effect – sensitivity is reduced in the neighbourhood of regions with large intensity variations.

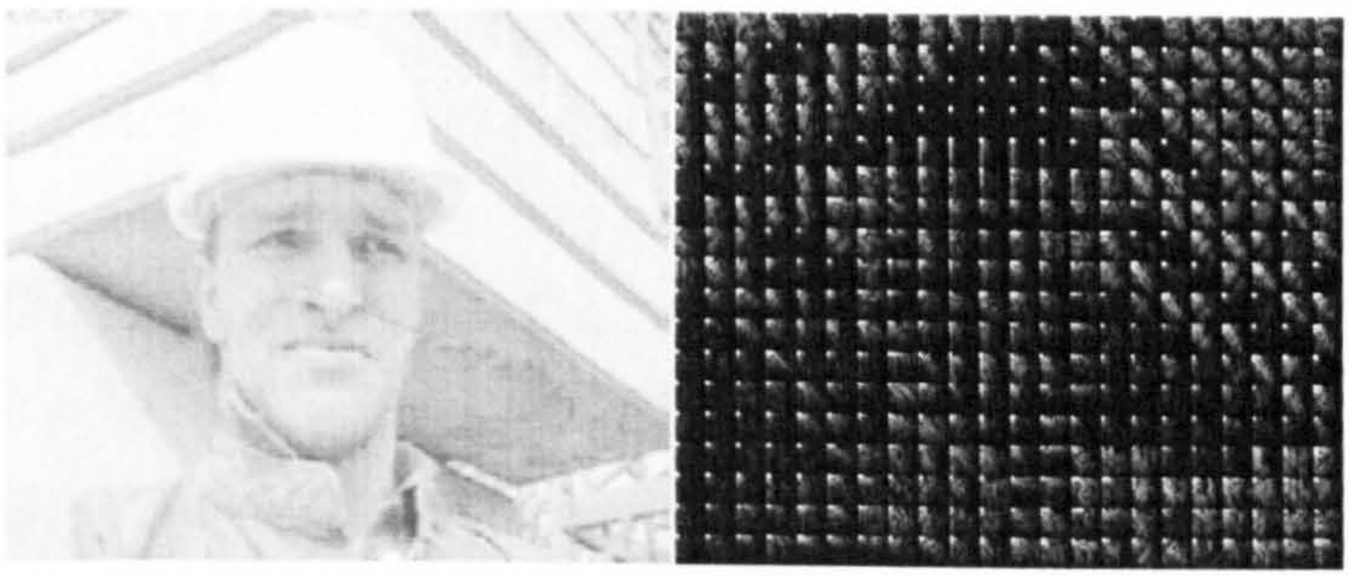
By exploiting the above properties, lossy schemes exploit our reduced ability to see detail immediately after a picture change, on the diagonal or in moving objects. Unfortunately, the latter doesn't yield as much of a savings as one might first think, because we often track moving objects on a screen with our eyes.

In addition to the response to intensity, another property of the HVS is concerned with colour response – the perception of fine colour details is limited compared with that of intensity. Hence chrominance resolution can be reduced by factors of two, four, eight or more, depending on the application. This gives rise to the 4:2:0, 4:2:2 and 4:1:1 subsampling schemes discussed in previous section.

2.3.3 Transform Coding

Video compression also relies heavily on the correlation between adjacent pixels. The spatial differential entropy in the previous section reveals a substantial amount of spatial redundancy. Predictive coding encodes pixel values in terms of the difference between the current pixel value and a predicted value from the causal neighbourhood. Another way of removing spatial redundancy is by transforming the pixel data into another domain whose coefficients are distributed more favourably for compression. The distribution can have lower entropy, or the energy can be compacted in just a few coefficients. A common transform used in video coding is the Discrete Cosine Transform (DCT).

Predictive coding relies on making an estimate of the current pixel from the previous values for that location and other neighbouring areas. The rules of this prediction are intrinsic to the encoder and decoder system and are hence known a-priori to the decoder. For any new pixel, the encoder need only send the difference or error value between what the rules would have predicted and the actual value of the new element. The more accurate the prediction, the less data needs to be sent.



(a) (b)

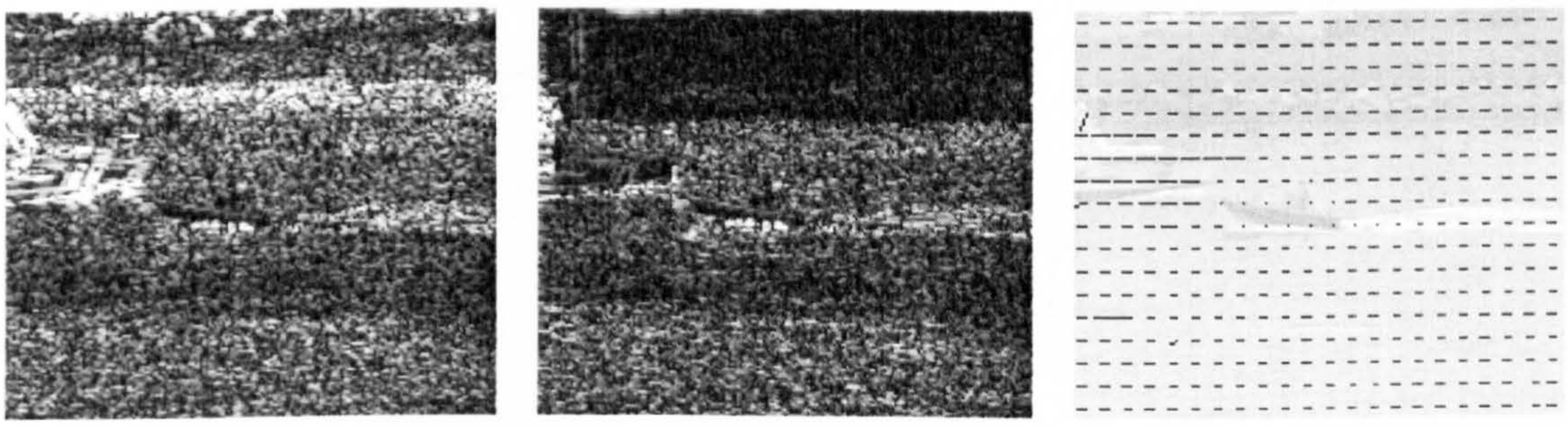
Figure 2.10 A frame in FOREMAN.QCIF sequence (a) and magnitude of its 8x8 block DCT coefficients (b). Entropies of (a) and (b) are 7.29 bits and 4.87 bits respectively.

From Figure 2.10, the DCT energies are compacted into the low frequency coefficients. It is also noted that the oblique edges in the top background manifest as blocks with non-zero higher frequencies coefficients. Coefficients at the lower right portions of the picture are more sparsely distributed as there is more variation in the pixel data around the region. Since a major part of the picture is uniform and pixels do not change abruptly, transform coefficients are very compact. A comparison of entropies between pixel data and the transform coefficients in Figure 2.10 shows a reduction of approximately 2.42 bits by DCT.

2.3.4 Motion Estimation and Compensation

The motion of objects or the camera from one frame to the next complicates predictive coding, but it also opens up new compression possibilities. Fortunately, moving objects in the real world are somewhat predictable. They tend to move with inertia and in a continuous fashion. By finding correspondence of the regions between frames, predictive coding can be applied in the temporal domain. The process is typified by finding the apparent motion of objects in the scene, hence the term motion estimation. Motion estimation has been known to be the main contributor to the large compression ratio of video coders. The down side of motion-compensated compression is the inter-frame dependency. Effects of errors induced in one frame gets propagated to subsequent frames. As a result, independently coded pictures must be used at regular intervals to remove such dependencies.

The correspondence between two or more frames in the picture sequences are widely exploited in both video compression and analysis applications. In video compression, frame differencing (FD) is the simplest means to remove temporal redundancy to achieve higher compression ratios. However, inter-frame relation between co-located pixels only holds if the objects remain stationary; object motion reduces this inter-frame correlation. A better alternative to simple frame-differencing is motion estimation: by matching each pixel with a neighbouring pixel from other frames which gives the best match, a displacement is found and the resulting residue is the difference between the current pixel and the best-matched pixel from another frame. The collection of displacements from all pixel is termed as the displacement field or motion vector field (MVF); the resulting pixel residues form the displaced frame difference (DFD). The dense MVF requires a fair amount of bits to code; it is commonly replaced by a sparse version in which groups of pixels are represented by a single motion vector as depicted in Figure 2.11.



(a) (b) (c)

Figure 2.11 An illustration of advantage of local motion estimation and displaced frame difference: (a) shows an absolute frame difference (|FD|); (b) shows the absolute displaced frame difference (|DFD|) and (c) shows the motion vector field. Comparison of (a) and (b) reveals that motion estimation reduces inter-frame redundancy better than simple frame differencing.

The DFD in (b) shows a marked reduction in the residual energy compared with the FD in (a). The ship at the left hand side is correctly predicted by motion estimation. This thesis is mainly concerned with improvements on existing motion estimation algorithms and proposal of novel algorithms for representing such fields. In additional a better match can be obtained by 'warping' the reference frame, either by a single motion parameter (as in global motion estimation) or a few parameters, each on a separate region (as in motion segmentation). The warping of the reference frame gives a better predictor and hence reduces residual entropies.

2.3.5 Quantization – Lossy Compression

When perceptual redundancy and all statistical redundancies (spatial, temporal and those due to distribution of the codewords) have been fully exploited, the only means of further compression is to remove some information by means of quantization. In general, quantization is the process where a set of data is represented by a reduced set of symbols. As a result of quantization, a range of data symbols are represented as a single symbol and consequent reproduction would only yield an approximation of the original data. It is this approximation that accounts for distortion caused by quantization.

We have cited earlier an example of quantization, which is the reduction of least significant bits in pixel value. Quantization where a single data value is coded as a symbol representative of a range of values is called scalar quantization. A representative scalar quantization scheme is the linear quantization where the quantized value is equally spaced in the domain and the ranges of every quantizer value are equal.

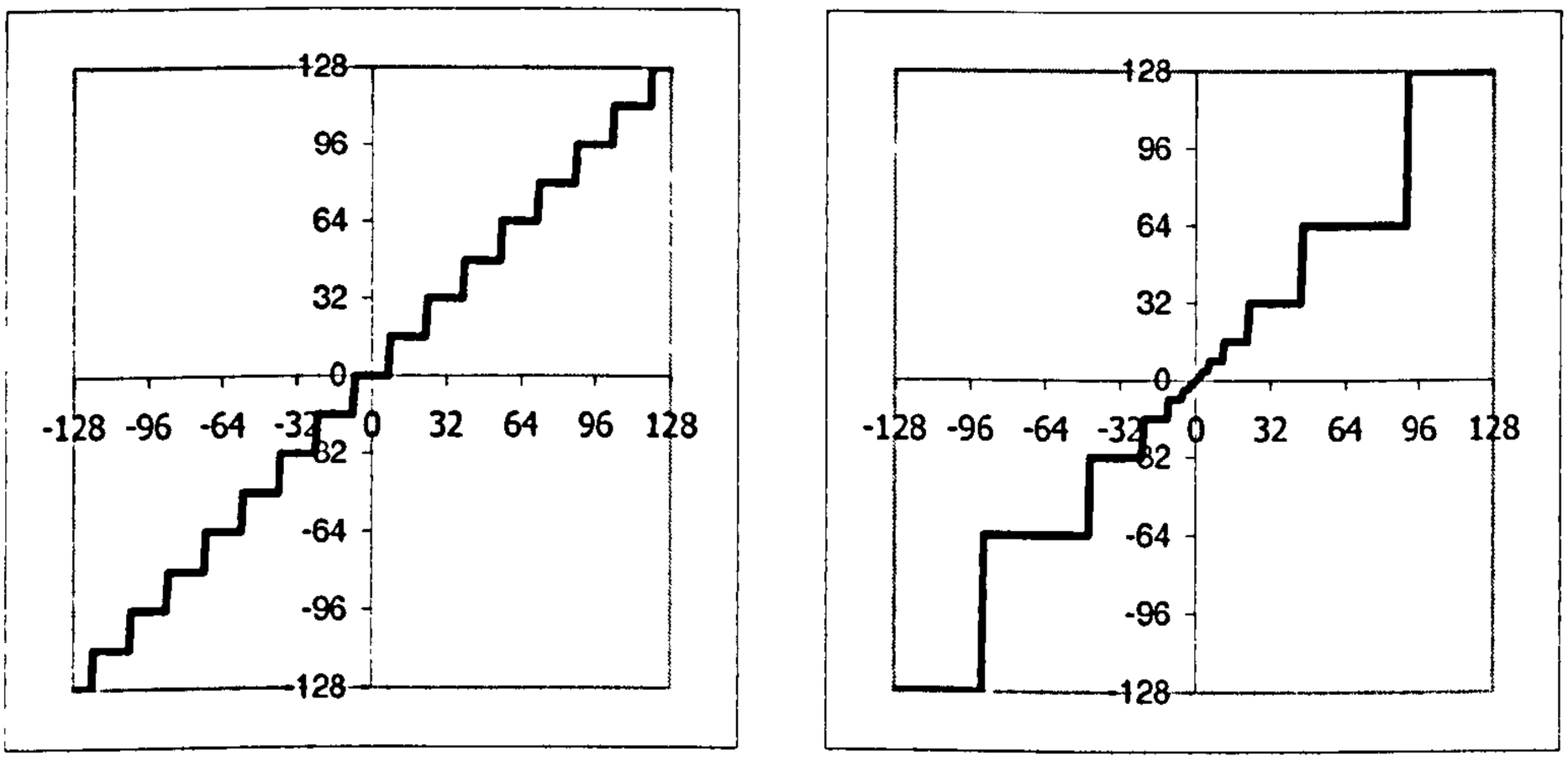


Figure 2.12 Quantizer transfer function, where x-axis is the input and y-axis is the representative value. (a) Linear quantization; (b) Non-linear quantization (exponential)

As illustrated in Figure 2.12, scalar quantization converts a single data value to its corresponding quantized value. Vector quantization, on the other hand, maps a group of data into a vector and assigns another vector from a reduced set of code vector which best matches the original vector. An example of vector quantization is colour quantization as shown below where C_b and C_r are quantized jointly.

After describing the general classes of compression techniques, the next section summarizes the past and state-of-the-art compression standards commonly used in the industry.

2.4 Video Compression Standards

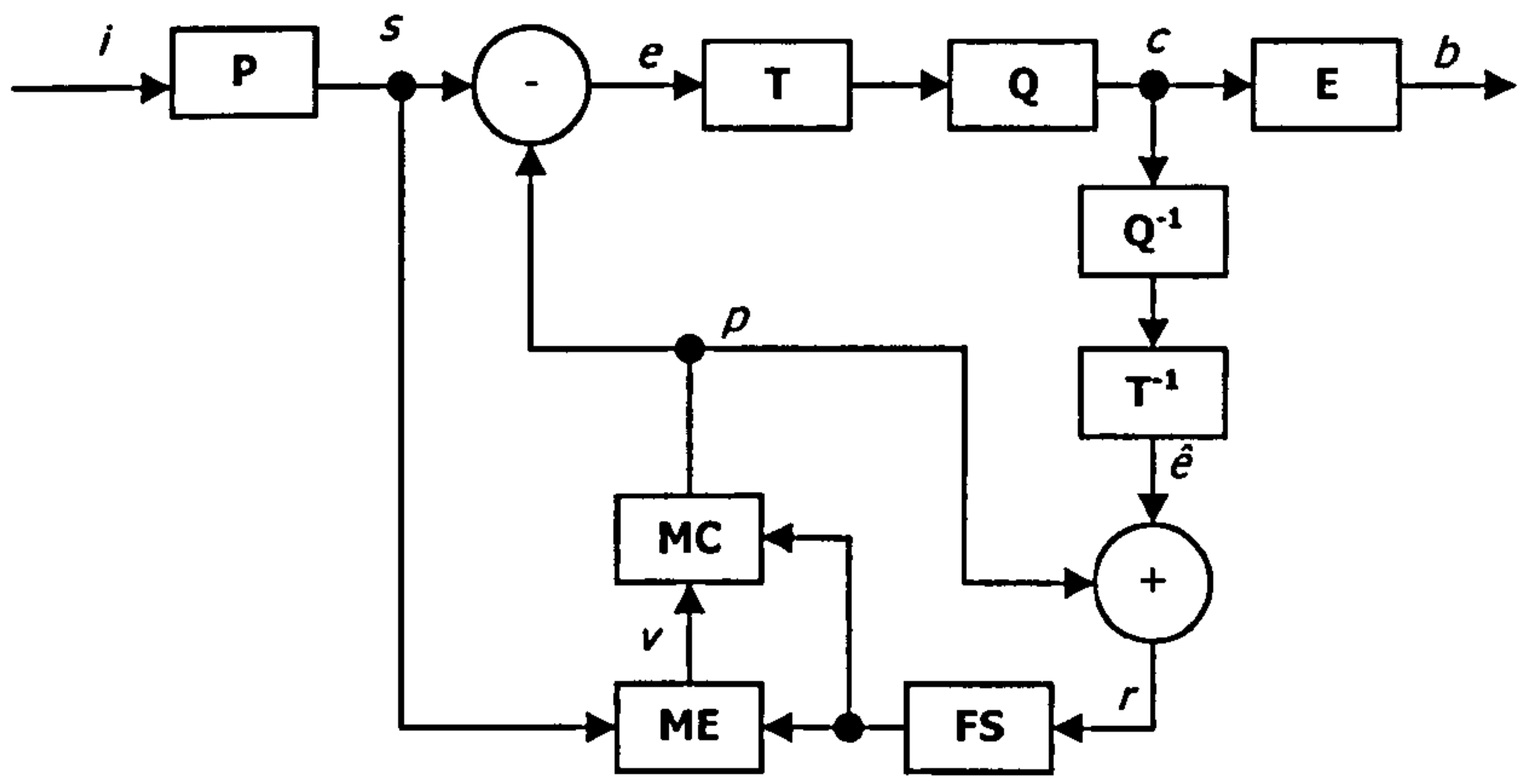
Standardized coding for system inter-operability is the main necessity for widespread deployment of video communication and storage services [Has-94]. Since the dawn of digital video processing in the nineteen-seventies, ISO/IEC and ITU has proposed various compression standards. These standards undergo substantial changes due to the technological advancements which provide ever-increasing processing power required by the video processors, and the rapid changes in demands from the, mass-consumer, military and medical sectors. Table 2.4 gives a summary of what these standards are and what applications they are targeted at:

Table 2.4 Various video coding standards and their applications.

Standard	Date/Author	Resolution Supported	Data Rates	Applications
H.261	1990,ITU	QCIF, CIF	40 Kbps – 2 Mbps	ISDN-based video conferencing
MPEG-1	1993, ISO/IEC	352x240, 352x288	<1.5 Mbps	VideoCD
H.263	1995, ITU	SQCIF, QCIF, CIF, 4CIF, 16CIF	<2 Mbps, more effective at <64 Kbps	VLBR video over POTS, GSM, video conferencing,
MPEG-2	1994, ISO/IEC	720x576 or below	<15 Mbps	DVD
MPEG-4	1998, ISO/IEC	Highly scalable	Highly scalable	Digital television, interactive graphics applications, interactive multimedia internet distribution.
H.264	TBD, ITU	Highly scalable	Highly scalable	Next generation codec for all purposes.

2.4.1 The generic Encoder

Despite the numerous video coding standards, all are based on general structure as shown in Figure 2.13.



Key (blocks):

P = Partitional
T = Transform Coder
Q = Quantiser
Q^{-1} = Dequantizer
T^{-1} = Inverse Transform
FS = Frame Store
ME = Motion Estimator
MC = Motion Compensator
E = Serializer + entropy coder.

Key (signals):

i = input frame
e = displaced frame diff (DFD)
p = predicted frame
$\hat{e}$ = reconstructed DFD
r = reconstructed frame
s = partitioning information
b = bit-stream
v = motion vector field
c = quantized coefficients

Figure 2.13 The generic framework of video compression system.

In the typical video encoder, the input frame *i* is partitioned into manageable blocks in the partitioning functional block **P** for further processing. Usually the luminance picture is partitioned into 16×16 blocks. Taking the mandatory 4:2:0 format supported by all standards, the two chrominance frames have partitions of 8×8 blocks. The luminance block is commonly subdivided further into four 8×8 sub-blocks. Along with the two chrominance blocks, the six blocks forms the basic coding structure of the typical encoder, usually called the Macro-block, as shown in Figure 2.14.

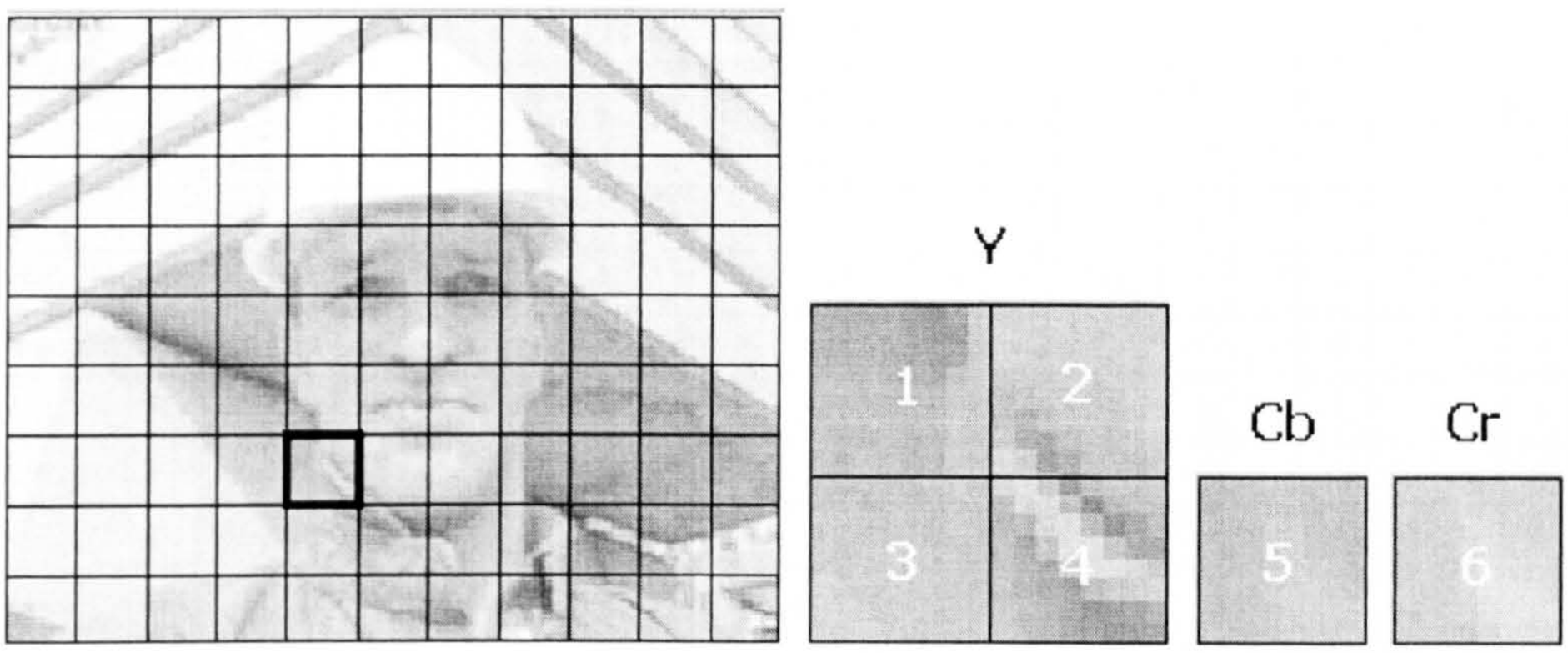


Figure 2.14 An illustration of a macroblock. The left-most picture shows a QCIF-4:2:0 picture partitioned into 16x16 arrays of macroblocks. One macroblock consists of four luminance(Y) blocks and 2 chrominance (C_b and C_r) blocks, sequenced in the numbered order.

Redundancy within a frame is reduced by transform coding, usually with the 8×8 discrete cosine transform DCT, although the discrete wavelet transform (DWT) and the 4×4 integer-based Hadamard transform is gaining popularity in the more recent standards (MPEG-4 and H.264 respectively). Due to the high correlation of neighbouring pixels, representing pixel information within a block in the frequency domain has the advantage that the majority of the high frequency coefficients will be zero. By suitably scanning the coefficients in ascending order of component frequency (a zigzag scan), a string of running zeroes interrupted occasionally by a non-zero number can be obtained. The string can then be easily encoded (usually run-length coding followed by entropy coding), as in the functional block **E** of Figure 2.13. Higher compression rates can be achieved if the coefficients are quantized before serializing, which creates more zero-runs and smaller non-zero values. This is the lossy compression carried out in block **Q**. The combined operation of **T, Q** and **E** is illustrated in Figure 2.15. The $\mathbf{T}^{-1}$ and $\mathbf{Q}^{-1}$ blocks are the counter part of **T** and **Q** respectively, whose purpose is to reconstruct the block $\hat{e}$ which is the exact copy of what the decoder produces. Due to the quantizer, the reconstructed signal $\hat{e}$ is a distortion of the original input signal e.

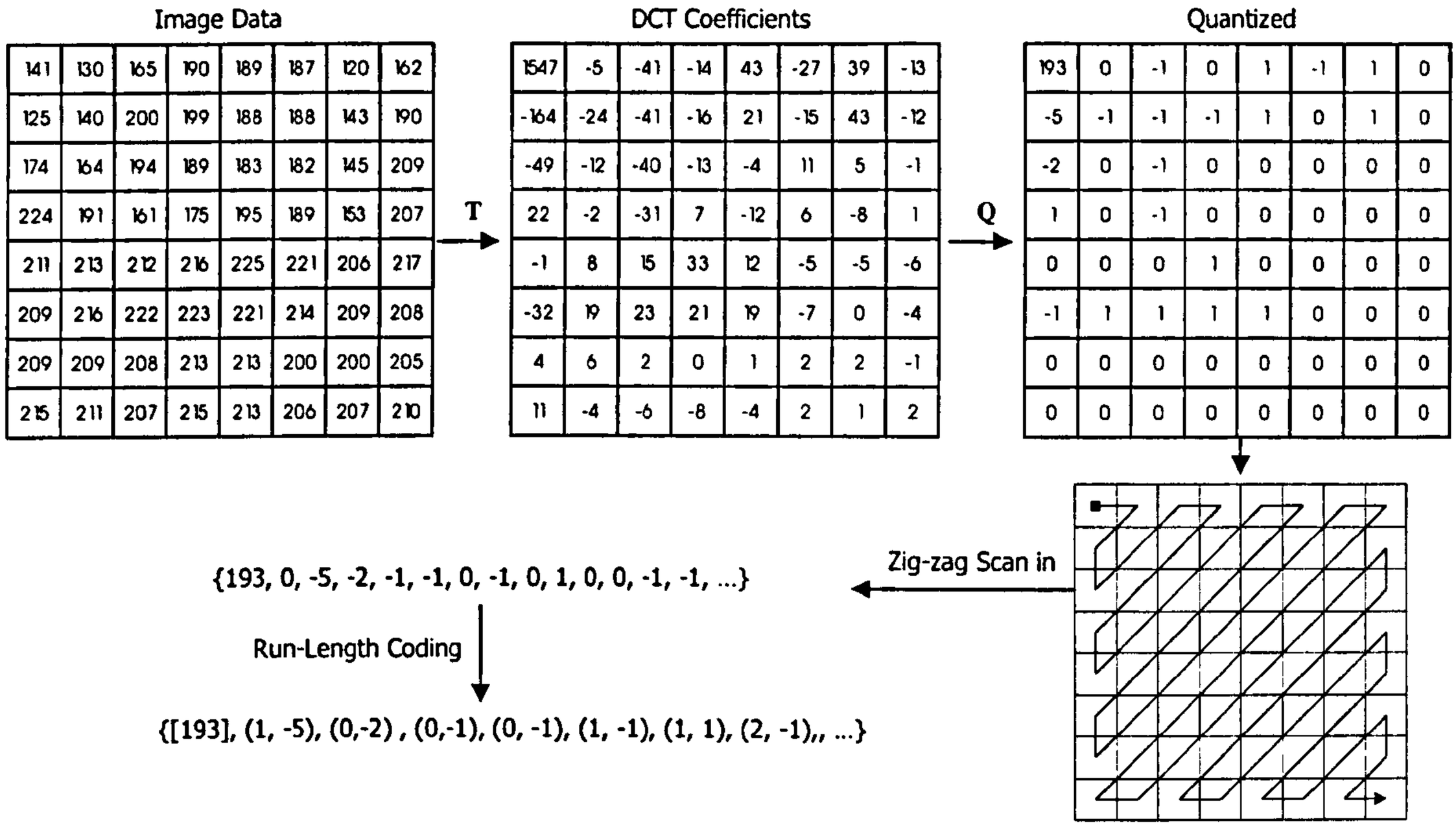


Figure 2.15 An illustration of DCT, quantization, zigzag scan and run-length coding.

The remaining three blocks, the frame-store (**FS**), the motion estimator (**ME**) and the motion compensator (**MC**) are involved with removing inter-frame redundancy. The frame store acts as a memory storage for previous reconstructed frames *r*, which is used as reference frames for motion estimation in block **ME**. The **ME** block takes the current block *s*, and compares the surrounding of the reference frame for the best match. The relative location of this best match with respect to the location of the current block forms the motion vector information, *v*. The motion compensator **MC**, extract the *v*-displaced block from the reference frame to form the predictor block, *p*. The predictor block is subtracted from the input block to form the displace frame difference *e*, which is fed into the transform coder (**T**) and the whole process repeats. In the case of intra-coding, *p* is set to zero so *e* is simply the input block.

The following sections describe briefly the historical and functional aspects of various coding standards.

2.4.2 H.261

The H.261 standard was the first video compression standard recommended by the International Telecommunication Union (ITU) for videoconferencing applications over the ISDN [ITU-93]. The main framework used in H.261 is identical to that shown in Figure 2.13 and has been used throughout succeeding video coding standards up until now. The standard supports coding pictures in the CIF and QCIF formats and the basic coding structure is the Macro-block. H.261 has been used in the last two

decades of the twentieth century. It has since been slowly superseded by the newer, more efficient standards – H.262 (MPEG-2) and H.263 standards.

2.4.3 MPEG-1

The H.261 video compression standard is targeted at applications for real-time communications. About the same time H.261 was standardized, the Moving Pictures Expert Group (MPEG) from ISO was drafting a similar standard for off-line processing of video and audio data for storage in the entertainment applications. The result is the MPEG-1 standard in the early nineties. MPEG-1 and the subsequent MPEG-2 and MPEG-4 standards are multi-part standards; they encompass audio, video, system and other parts. We shall focus on the video coding part. Essentially, the coding MPEG-1 standard is similar to the H.261 standard. As MPEG-1 has to cater for the entertainment field, it must take formats compatible with both analog sources like NTSC and PAL, which makes the SIF formats a natural choice.

The MPEG-1 standard (formally known as ISO/IEC 11172) [MPE-93] provides superior compression to the H.261 standard with the some additional features. One major improvement is an addition of the bi-directionally predicted frame (B-frame) in which the current frame can be predicted from a previous and future frame. This implies a delay in between the reconstruction and display at the decoder, because the decoded sequence is not necessarily the display sequence (see Figure 2.16). Nevertheless, this is not a major disadvantage in playback applications considering the amount of compression deliverable by B-frames.

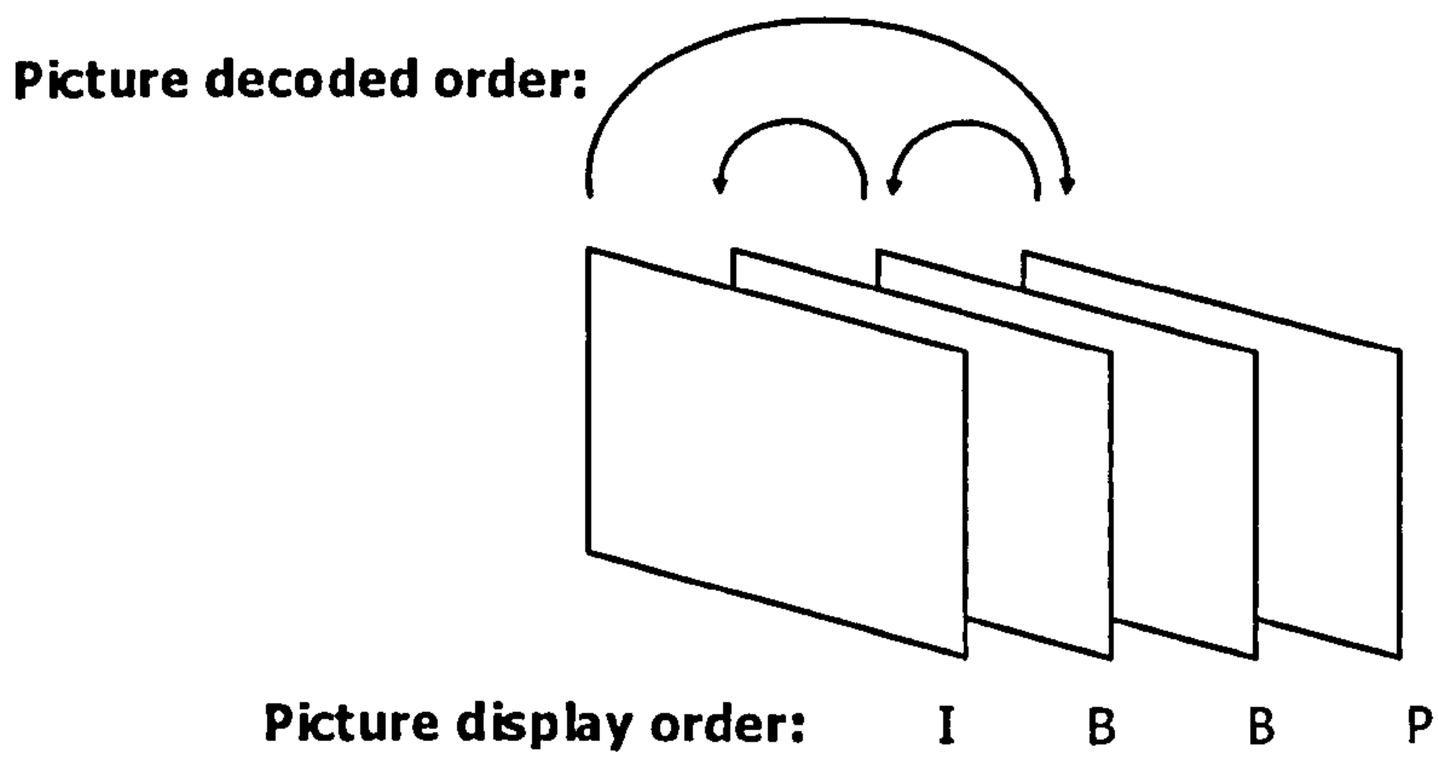


Figure 2.16 Difference in the picture decoded order and the display order due to B-frame coding in MPEG-1.

In the mass consumer market, MPEG's attempt of standardizing VHS-quality VideoCD via MPEG-1 standard has shown limited success, particularly in the low-end markets which is more tolerant to the blocking artefacts brought about by the block-based discrete cosine transform (DCT) and quantization.

It is its successor, MPEG-2, which makes cinematic experience possible in homes through DVDs by providing a more options in input formats and more sophisticated compression algorithms, as shown below.

2.4.4 MPEG-2

MPEG-2 [MPE-95] was produced in collaboration with ITU-T (which also published it as the H.262 standard). MPEG-2 is intended to provide compression for studio-quality video applications like digital broadcasting, DVD (digital versatile disks) and video-on-demand over ATM (Asynchronous Transfer Mode) networks. The standard was originally designed for high-quality encoding of interlaced video on standard TV with bit rates around 4 to 9 Mbps. MPEG then proceeds to draft the MPEG-3 standards for high-density TV (HDTV). The group later realised MPEG-2 is equally capable of handling such applications. MEPG-3 was subsequently disbanded with all the works absorbed into MPEG-2.

The main contribution of the MPEG-2 is the facility for video formats compatible with the analog formats (NTSC, PAL, SECAM, etc). All three colour sampling formats of 4:2:0, 4:2:2 and 4:4:4 are supported. Furthermore, both interlaced and progressive video are possible; motion estimation can be field-oriented for better compression.

Another important addition is the concept of scalability. In the scalability mode, each picture is sent in two layers:

1. Basic layer which carries essential data to reconstruct a low-quality picture.
2. Enhanced layer which carries additional information to improve the visual quality of current picture.

With scalability mode, it is possible for a basic decoder to extract a lower rate bitstream which will reproduce a lower quality image, while allowing an advanced decoder to receive the full resolution, high quality picture.

2.4.5 H.263/H.263+

At the same time ITU-T continues to improve its video-conferencing compression techniques to cater for the rates for the POTS networks, which brings about the H.263 standard [Rij-96] [ITU-98], subsequent improvements lead to the version 2 of the standard that provides more options and wider range of applications. It is commonly known as the H.263+ standard. The objective for H.263 and H.263+ was to provide significantly better quality then its predecessor H.261 for real-time transmission on networks with data rates below 64 kbps such as the PSTN and the GSM [Kar-96].

The basic building block of the H.263 encoder is similar to that of the H.261. Refer to [ITU-98] for a detailed description of the semantics of the bit-stream.

Motion compensation is done with reference to the previous frame and vectors are represented at half-pixel resolution. Motion compensation at ½-pixel resolution is by frame interpolation with the [¼, ½, ¼] bilinear filter. The VLC uses a novel 3-dimensional (LAST, RUN, LEVEL) run-length codes.

In addition to the basic coder specifications, the H.263+ standard provides various optional schemes to improve coding efficiencies at the expense of higher processing resource requirements. These options include the possibility of coding motion vectors outside the reference picture, overlapping of motion compensated blocks to reduce blocking artefacts. A new partitioning syntax called the slice is used to enable a more region-based approach to video coding.

Compared with the MPEG-2 standard, H.263+ provides a much more sophisticated scalability controls (spatial, temporal, and SNR scalabilities). The H.263+ is also better suitable to provide error resilience to noisy transmission channel and heterogeneous networks.

2.4.6 MPEG-4

The MPEG group designs MPEG-4 [Per-00] [Ava-00] to allow the user to interact with the objects in the scene within the limits set by the author. It also brings multimedia to low bit-rate networks [Sik-97] [ISO-98]. MPEG-4 uses media objects to represent aural, visual or audiovisual content. Media objects can be synthetic like in interactive graphics applications or natural like in digital television. These media objects can be combined to form compound media objects. MPEG-4 multiplexes and synchronizes the media objects before transmission to provide. MPEG-4 organizes the media objects in a hierarchical fashion where the lowest level has primitive media objects like still images, video objects, and audio objects. MPEG-4 has a number of primitive media objects that can be used to represent 2 or 3-dimensional media objects. MPEG-4 also defines a coded representation of objects for text, graphics, and synthetic sound, talking synthetic heads.

The visual part of the MPEG-4 standard describes methods for compressing images and videos, compressing textures for texture mapping of 2-D and 3-D meshes, compressing implicit 2-D meshes, and compressing time-varying geometry streams that animate meshes. It also provides algorithms for random access to all types of visual objects as well as algorithms for spatial, temporal and quality scalability, content-based scalability of textures, images and video. Algorithms for error robustness and resilience in error prone environments are also part of the standard. For synthetic objects MPEG-4 specifies parametric descriptions of human face and body, parametric descriptions for animation streams of the face and body. MPEG-4 also describes static and dynamic mesh coding with texture mapping, texture coding with view dependent applications. MPEG-4 supports coding of video objects with spatial and temporal scalability. Scalability allows decoding a part of a stream and construct images with reduced decoder complexity (reduced quality), reduced spatial resolution, reduced temporal resolution, or with equal temporal and spatial resolution.

2.4.7 H.264/MPEG-4 Part 10/AVC

After finalising the original H.263 standard for video telephony in 1995, the ITU-T Video Coding Experts Group (VCEG) started work on a "short-term" effort to add extra features to H.263 (resulting in Version 2 or H.263+). At the same time, a "long-term" effort is underway to develop a new standard for low bit rate visual communications called "H.26L" standard, offering significantly better video compression efficiency than previous ITU-T standards. In 2001, the ISO MPEG recognised the potential benefits of H.26L and the Joint Video Team (JVT) was formed, including experts from MPEG and VCEG. JVT's main task is to develop the draft H.26L "model" into a full International Standard. The outcome will be two identical) standards: ISO MPEG4 Part 10 of MPEG4 and ITU-T H.264. The "official" title of the new standard is Advanced Video Coding (AVC); however, it is widely known by its old working title, H.26L and by its ITU document number, H.264 [Wei-03] [JVT-02] [Ric-www].

Common to earlier standards (such as MPEG1, MPEG2 and MPEG4), the H.264 draft standard does not explicitly define a codec. Rather, the standard defines the syntax of an encoded video bitstream together with the method of decoding this bitstream. The basic functional elements (prediction, transform, quantization, entropy encoding) are little different from previous standards (MPEG1, MPEG2, MPEG4, H.261, H.263); the important changes in H.264 occur in the details of each functional element. The main improvements relevant to real-time applications include:

- 4x4 luma prediction modes – encodes INTRA 4x4 blocks by predictors of surrounding blocks.
- Variable block-size motion estimation – allows 16x16, 8x16, 16x8, 8x4, 4x8, 4x4 based motion estimation to better match blocks to true regions.
- 4x4 residual transform and quantization – integer-based joint transforms and quantization scheme to reduce processing requirements.
- Sub-pixel resolution motion estimation – allows 1/2-pixel and 1/4-pixel motion estimation for better temporal prediction.

Other features like B pictures and block-based multiple reference frames prediction are not feasible for real-time implementation and will not be discussed here. The reader can consult [JVT-02] for the full description of the coding standard.

2.5 Video and Image Segmentation

Other than MPEG-4, all prevailing video coding standards are based on partitioning frames into regular square grids. The historic H261 started the trend by introducing the macroblock concept where a picture is partitioned into 16x16 blocks and coded in a raster-scan order. MPEG-1 and MPEG-2 adopts the paradigm and H.263 started out similarly. A higher layer of coding is the GOB (group of blocks)

where macroblocks are grouped sequentially into horizontal strips. Other segmentation coding techniques have been extensively proposed and researched. They include quad-tree decomposition, edge-line coding and region adjacency graph coding. However, these methods are yet to be incorporated into any standards due to complexity issues and are generally application-specific. The state-of-the art till date remains at improving the macroblock structure:

- Coding of macroblock-based slices in arbitrary order.
- Alpha coding within a macroblock to represent different layers within a macroblock.

Despite being included in the standards, the MPEG-4 segmentation is rarely used in real-time coding applications. Until the complexity issues can be solved, simple segmentation of regular slices and GOB will prevail. On the H.264 front, main research efforts are spent on improving compression efficiencies in the macro-block level and segmentation coding has not been included. This provides an interesting area of research in the near future.

2.6 Summary and Comments

This chapter has reviewed the basic concepts of video coding together with a survey of current standards. It has explained the major issues associated with various sources of redundancies, and highlighted how these redundancies can be removed.

The main thrust of this thesis is to find ways to remove temporal redundancies by means of motion estimation and segmentation. The target is to provide the intra-coder (texture coding) with the displaced frame difference (DFD) (or textural reside) containing the least entropy. The principal concern in this thesis is lossless inter-frame processing. It is believed that providing a lowest entropy residual to the texture coder and quantizer is crucial to having a good rate-distortion curve as the amount of information to start with is minimized. Another part of the thesis is concerned with removing redundancy in the motion vector field without affecting the textural entropy. This is not related to the rate-distortion optimization problem, but is aimed at reducing the overhead of coding the motion information which constitute a higher proportion of the bit budget in recent coding standards where block sizes get smaller and target bit-rates gets lower.

Chapter 3:

Local Motion Estimation

The common notion of video as a sequence of images is an incomplete one. It fails to reflect the fact that the consecutive images in a sequence are highly correlated with each other most of the time. This correlation results in a high inter-image or inter-frame redundancy. By exploiting these redundancies, video sequences can be much more highly compressed than a collection of independent images. Motion estimation is the general term referring to the matching of pixels or groups of pixels between two or amongst more frames to remove these redundancies for coding purposes. On the other hand, motion estimation is also a low-level analysis tool to extract and track moving objects from a video sequence. Local motion estimation (LME) refers to a group of algorithms where pixels or small groups of neighbouring pixels are matched to produce a map of correspondences between two or more frames. This is in contrast with global motion estimation (GME), where a frame is motion estimated as a whole, or as a segmentation of a few large regions which is deemed to have been transformed and moved as a contiguous entity. This chapter and the next (3 and 4) focus on local motion estimation whilst in the following two chapters (5 and 6), global motion estimation is discussed.

Chapter 3 will be dedicated to basic principles and prior art, where different approaches to LME will be discussed. One method in particular, the block matching algorithm (BMA in short), will be singled out as an algorithm of choice due to its ease of implementation and computational tractability. In chapter 4, a few novel insights into BMA will be presented. New algorithms of BMA that incorporate elements from other motion estimation approaches will also be introduced.

3.1 Basic Principles of Motion Estimation

Motion estimation is the determination of the correspondence between pixels in one picture and pixels in another. In this thesis, we focus on estimating the motion of current frame $I(x, y, t)$ with respect to the immediate previous frame $I(x, y, t-1)$. Alternative schemes of using multiple reference frames and bidirectional prediction using future as well as previous frames can be easily adapted. The fundamental principle behind motion estimation is that pixel intensity $I(x, y, t)$ within an object does not change as the object moves. This is translated into the following differential equation:

$$\frac{dI(x,y,t)}{dt}=0 \qquad \text{Eq 3-1}$$

Taking partial derivatives in Eq 3-1 results in the optical flow equation (OFE) first used by Horn and Schunck [Tek-95] [Hor-81]:

$$\frac{\partial I}{\partial x}\frac{dx}{dt}+\frac{\partial I}{\partial y}\frac{dy}{dt}+\frac{\partial I}{\partial t}=0 \qquad \dots \text{(a)} \qquad \text{Eq 3-2}$$

$$I_x u + I_y v + \dot{I} = 0 \qquad \dots \text{(b)}$$

$$\nabla I \cdot \dot{\mathbf{p}} + \dot{I} = 0 \qquad \dots \text{(c)}$$

$$\nabla I \cdot \mathbf{v} + \dot{I} = 0 \qquad \dots \text{(d)}$$

In Eq 3-2 (b), (u, v) denotes the motion vector components; I_x, I_y and $\dot{I}$ are the partial derivatives of $I(x,y,t)$ with respect to the horizontal spatial, vertical spatial and temporal variables respectively. The dots in Eq 3-2 (c) represent partial derivatives with respect to t, and $\mathbf{p}$ represents pixel co-ordinate (x, y). Hence the derivative of $\mathbf{p}$ is the motion vector $\mathbf{v}$ in Eq 3-2 (d). The problem of motion estimation is thus the evaluation of $\mathbf{v}(\mathbf{p})$ for each $\mathbf{p}$ in which Eq 3-2 is satisfied.

Empirically, a pixel (moving or stationary) seldom maintains constant intensity across frames. Hence, the right hand side of Eq 3-1 is seldom zero. The reasons are many-fold. They include:

- Impulsive noise caused by the image capture process.
- Change in illumination on the object.
- Shadow cast on the object by other moving objects.
- Change in exposure settings of the capturing device.

Due to these errors, an alternative approach is to derive a discrete form for the left-hand-side of Eq 3-1, arriving at the equation for displaced frame difference (DFD) [Net-79] [Wal-84] [Tek-95]:

$$e(\mathbf{p},t) = I(\mathbf{p},t) - I(\mathbf{p} - \mathbf{v}(\mathbf{p},t), t-1) \qquad \text{Eq 3-3}$$

With Eq 3-3, the expression $\mathbf{v}(\mathbf{p})$ suggests a velocity as a function of location, thus forming the motion vector field. Then motion estimation is achieved by minimizing a specified norm $\|e(\mathbf{p},t)\|$:

$$\mathbf{v}(\mathbf{p},t) = \arg\min_{\mathbf{u}} \left\| I(\mathbf{p},t) - I(\mathbf{p}-\mathbf{u},t-1) \right\| \qquad \text{Eq 3-4}$$

Pixel-based minimization of Eq 3-4 usually produces poor results due to the sensitivity of the algorithm towards noise. Current and past video compression standards [ITU-93] [ITU-98] [MPE-93] [MPE-95] adopt a blocked-based minimization (block matching algorithm, BMA), thus producing a sparse motion vector field, which is both computationally tractable and robust.

Instead of minimizing the DFD norm, we can find the best match of the neighbourhood of (x, y) in frame t with a displaced neighbourhood in frame t-1 by means of the correlation function:

$$C(x,y,\Delta x,\Delta y,t_1,t_2) = \sum_{(i,j)\in N(x,y)} I(i,j,t_1) I(i-\Delta x, j-\Delta y, t_2) \qquad \text{Eq 3-5}$$

$N(x,y)$ is the set of points in a pre-defined neighbourhood of (x,y) including (x,y) itself. The motion vector (u,v) at location (x,y) would be the $(\Delta x,\Delta y)$ pair which produces the highest correlation or best match according to some similarity measure:

$$<u,v> = \arg\max_{(\Delta x,\Delta y)} C(x,y,\Delta x,\Delta y,t,t-1) \qquad \text{Eq 3-6}$$

Variations of Eq 3-6 has been used in several motion estimation algorithms based on the spatial frequency domain of (x,y,t), which has been claimed to be more precise in locating the vectors and is more robust towards noise due to variation in lighting conditions [For-02].

Next, we introduce statistical methods for motion estimation. These methods find the motion vector fields $\mathbf{V} = \{\mathbf{v}(\mathbf{p}); \in \Lambda\}$where Λ denotes the set of pixels in the frame, or the frame lattice. Given the current and previous frame in as intensity fields ($\mathbf{I}_t$ and $\mathbf{I}_{t-1}$) using maximum a-posteriori probability (MAP) estimation, a widely used Bayesian method [Leo-99] is given by:

$$p(\mathbf{V} \mid \mathbf{I}_t, \mathbf{I}_{t-1}) = \frac{p(\mathbf{I}_t \mid \mathbf{V}, \mathbf{I}_{t-1}) p(\mathbf{V} \mid \mathbf{I}_{t-1})}{p(\mathbf{I}_t \mid \mathbf{I}_{t-1})} \qquad \text{Eq 3-7}$$

The left hand side is the a posteriori probability, the probability of a certain motion vector field **V**, given the current frame and the previous frame. The first term in the numerator of the right-hand-side of Eq 3-7 is the likelihood of **V** giving rise to $\mathbf{I}_t$. The second term is the a-priori probability of the motion vector field, **V**, denoting how likely **V** is to occur, this is usually based on some spatial

distributional properties of the field. Motion estimation is the maximization of this MAP with respect to all or a subset of the **V** configurations. As the denominator in Eq 3-7 is independent of **V**, the MAP estimation can be simplified to:

$$\mathbf{V} = \arg\max_{all\ \mathbf{U}} p(\mathbf{U}|\mathbf{I}_t, \mathbf{I}_{t-1}) = \arg\max_{all\ \mathbf{U}} p(\mathbf{I}_t|\mathbf{U}, \mathbf{I}_{t-1}) p(\mathbf{U}|\mathbf{I}_{t-1}) \quad \text{Eq 3-8}$$

Note that **V** is a candidate of the whole motion vector field configuration, not an individual motion vector. The various motion estimation algorithms based on Bayesian methods vary in how the likelihoods and the priors are modelled. Regardless of the models chosen, maximization through a full search of the whole configuration space is too computationally intensive; various methods of sub-optimal search will be discussed in a subsequent section of this chapter.

All methods described above are applicable to pixel based algorithms, giving rise to a dense motion vector field. They are useful for extracting objects for advanced object-based coding, but the dense motion vector field needed as side information for coding proves to be too excessive to be transmitted or stored in video compression systems. The alternative for general video coding is to sub-sample the vector field and transmit the sparse motion vector field instead. This gives rise to region-based motion estimation.

The simplest form of region-based matching method uses $N_1 \times N_2$ non-overlapping rectangular blocks (square blocks are a special instance where $N_1 = N_2 = N$) from $I(x, y, t)$ and finds a block with the neighbourhood in $I(x, y, t-1)$ with the best match, each neighbourhood block is characterised by the offset $(\Delta x, \Delta y)$ between the neighbouring block and the current block, and the collection of search offsets forms the search window. Typical search window is the rectangular range $[-R, R]^2$ where R is a positive integer. This method results in a sparse $C \times R$ motion vector field from a $W \times H$ picture, where $C = W / N_1$ and $R = H / N_2$. The block matching algorithm (BMA) is used in virtually all video compression standards like MPEG1, MPEG2, MPEG4, H.261, H.263 and H.264.

$$\langle u, v \rangle^*_{r,c} = \arg\min_{\langle u,v \rangle \in W} \sum_{y=0}^{N_2-1} \sum_{x=0}^{N_1-1} \left\| I(cN_1 + x, rN_2 + y, t) - I(cN_1 + x + u, rN_2 + y + v, t-1) \right\| \quad \text{Eq 3-9}$$

Different BMA methods vary according to their:

- Matching criteria – the cost function $\|\bullet\|$, examples include sum-of-squared-difference (SSD), sum-of-absolute-difference (SAD), correlation-related measure, etc.
- Search strategies – the set of motion vectors to be searched, and the sequence of the

search procedure, examples are the full search [Bru-01], fast-full-search [Kog-81], three-step-search [Iai-81], 2-D logarithmic search [Sri-85], one-at-a-time-search [Li-94], new-three-step-search [Liu-96], block-based gradient descent search [Orc-94], etc.

- Block sizes – the values of N_1 and N_2, for instance 4x4 pixels, 8x8 pixels, 16x16 pixels, and varying block-sizes.

Extending BMA beyond regular blocks results in variable-block sized BMA and region-based motion estimation. The former usually involves quad-tree decomposition; the latter involves motion segmentation, which will be elaborated in later chapters.

The next section provides a detailed study of the prior art to these motion estimation methods.

3.2 Existing Methods of Estimating Local Motion

3.2.1 Methods using Optical Flow Equation

The optical flow equation (OFE) of Eq 3-2 contains 2 variables u and v for each pixel, requiring additional constraints if it were to be solved. The standard approach for handling this ill-conditioned problem is to assume that the motion vector field obeys certain spatial pattern constraint. Solution of the optical flow equation then involves the minimization of the combined effects of the OFE and a regularization term $E_s(u,v)$:

$$E(x,y) = \left[I_x(x,y)u(x,y) + I_y(x,y)v(x,y) + I_t(x,y)\right]^2 + \alpha E_s\left[u(x,y), v(x,y)\right] \qquad \text{Eq 3-10}$$

The value of α is used to adjust the relative weights of the OFE term and the constraint term.

The most common formulation of E_s is the first-order, or membrane, model [Tek-95]. This is based on the assumption that motion of individual pixels do not vary significantly across the region:

$$E_s^2(u,v) = \|\nabla u\|^2 + \|\nabla v\|^2 \qquad \text{Eq 3-11}$$
$$\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial y}\right)^2 + \left(\frac{\partial v}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2$$

In their classic paper, Horn and Schunck [Hor-81] used the following iterative equations to minimize $E(x,y)$:

$$u^{k+1} = < u^k > - \frac{I_x \left[I_x < u^k > + I_x < v^k > + I_t \right]}{\alpha^2 + I_x^2 + I_y^2}$$

$$v^{k+1} = < v^k > - \frac{I_x \left[I_x < u^k > + I_x < v^k > + I_t \right]}{\alpha^2 + I_x^2 + I_y^2}$$

Eq 3-12

In Eq 3-12, k is the iteration step number, $< u^k >$ and $< v^k >$ the neighbourhood averages of u^k and v^k.

The smoothness measure imposed by Horn and Schunck's method tends to blur out true motion boundaries and does not help in determining previously occluded regions. Nagel and Enkelmann [Nag-86] proposed the concept of directional smoothness constraints, which weighs the motion vector smoothness constraint according to the texture edges.

$$E_{ds}^2(u,v) = (\nabla u)^T \mathbf{W} (\nabla u) + (\nabla v)^T \mathbf{W} (\nabla v)$$

Eq 3-13

3.2.2 Pel-Recursive Methods with Displaced Frame Difference

The pel-recursive method aims to minimizing the energy of displaced frame difference in Eq 3-4:

$$e^2(x,y,t) = \left[I(x,y,t) - I(x - u(x,y), y - v(x,y), t-1) \right]^2$$

Eq 3-14

There are two common approaches – the steepest-descent method and Newton-Raphson method [Pre-02]. The steepest-descent method moves the candidate estimator away from the direction of the gradient, as in Eq 3-15. The equation is a pixel-wise expression with $\mathbf{u}$ is the motion vector (u,v) and $E(\mathbf{u})$ has its implicit variable x, y, t for clarity; k is the iteration number. The value of α is chosen to be small enough to prevent oscillation, yet large enough to ensure rapid convergence.

$$\mathbf{u}_{k+1} = \mathbf{u}_k - \alpha \nabla_{\mathbf{u}} E(\mathbf{u}) \big|_{\mathbf{u}_k}$$

Eq 3-15

$$\nabla_{\mathbf{u}} E(\mathbf{u}) = \begin{bmatrix} \frac{\partial E}{\partial u} \\ \frac{\partial E}{\partial v} \end{bmatrix}$$

The Newton-Raphson method is used to find the root of $E'(\mathbf{u}) = 0$:

$$\mathbf{u}_{k+1} = \mathbf{u}_k - \mathbf{H}^{-1}\nabla_{\mathbf{u}} E(\mathbf{u})\big|_{\mathbf{u}_k} \qquad \text{Eq 3-16}$$

$$\mathbf{H} = \begin{bmatrix} \frac{\partial^2 E}{\partial x^2} & \frac{\partial^2 E}{\partial x \partial y} \\ \frac{\partial^2 E}{\partial y \partial x} & \frac{\partial^2 E}{\partial y^2} \end{bmatrix}$$

Netravali and Robbins [Net-79] first used the steepest descent method by replacing the gradient term with a calculable form:

$$\mathbf{v}_{k+1}(\mathbf{p}) = \mathbf{v}_k(\mathbf{p}) - \varepsilon\left[I(\mathbf{p},t) - I(\mathbf{p} - \mathbf{v}_k(\mathbf{p}), t-1)\right]\nabla_{\mathbf{p}} I(\mathbf{p} - \mathbf{v}_k(\mathbf{p}), t-1) \qquad \text{Eq 3-17}$$

Walker and Rao [Wal-84] attempt to improve Netravali-Robbin's algorithm by using adaptive step size: ε

$$\varepsilon = \frac{1}{2\left\|\nabla_{\mathbf{p}} I(\mathbf{p} - \mathbf{v}, t-1)\right\|^2} \qquad \text{Eq 3-18}$$

Cafforio and Rocca [Caf-83] used an improved version of Eq 3-18 by adding a constant term to prevent division by zero:

$$\varepsilon = \frac{1}{\left\|\nabla_{\mathbf{p}} I(\mathbf{p} - \mathbf{v}, t-1)\right\|^2 + \eta^2} \qquad \text{Eq 3-19}$$

3.2.3 Bayesian Methods

As discussed in the last section, motion estimation can be viewed as finding the motion vector field modelled as a random field which gives rise to the maximum a-posteriori probability given the current and previous frames. Bayesian methods are composed of two parts:

1. Modelling of the likelihood and the a priori fields.
2. Maximization of the MAP which is expressed as the product of the likelihood and prior fields.

In its simplest form, the likelihood field is modelled by assuming that the change in the intensity of the pixel along the true motion trajectory is due to observation noise. By assuming this noise to be a zero mean Gaussian distribution with variance σ^2 identically and independently distributed across all pixels, we arrive at the likelihood model [Tek-95], in terms of the displaced frame difference:

$$p(\mathbf{I}_t \mid \mathbf{V}, \mathbf{I}_{t-1}) = \frac{1}{|\Lambda|\sqrt{2\pi\sigma^2}} \exp\left(- \sum_{\mathbf{p}\in\Lambda} \frac{[I(\mathbf{p},t) - I(\mathbf{p}-\mathbf{v}(\mathbf{p}),t-1]^2}{2\sigma^2} \right) \quad \text{Eq 3-20}$$

To model the prior probability, we first observe that in most scenes motion is the result of movements of rigid objects. The motion field within each object should be quite uniformly changing, if not similar. Hence, it is usual to impose a smoothness constraint to the motion vector field, which assumes that motion vector fields are smooth except at object boundaries. Based on this assumption, the motion vector field prior can be modelled as a Markov random field [Li-00]. A Markov random field is a random field whose probability of a site (in the case of motion vector field, the probability of a pixel having a certain motion vector) is conditional upon the close vicinity (neighbourhood) of the site, as illustrated in Figure 3.1.

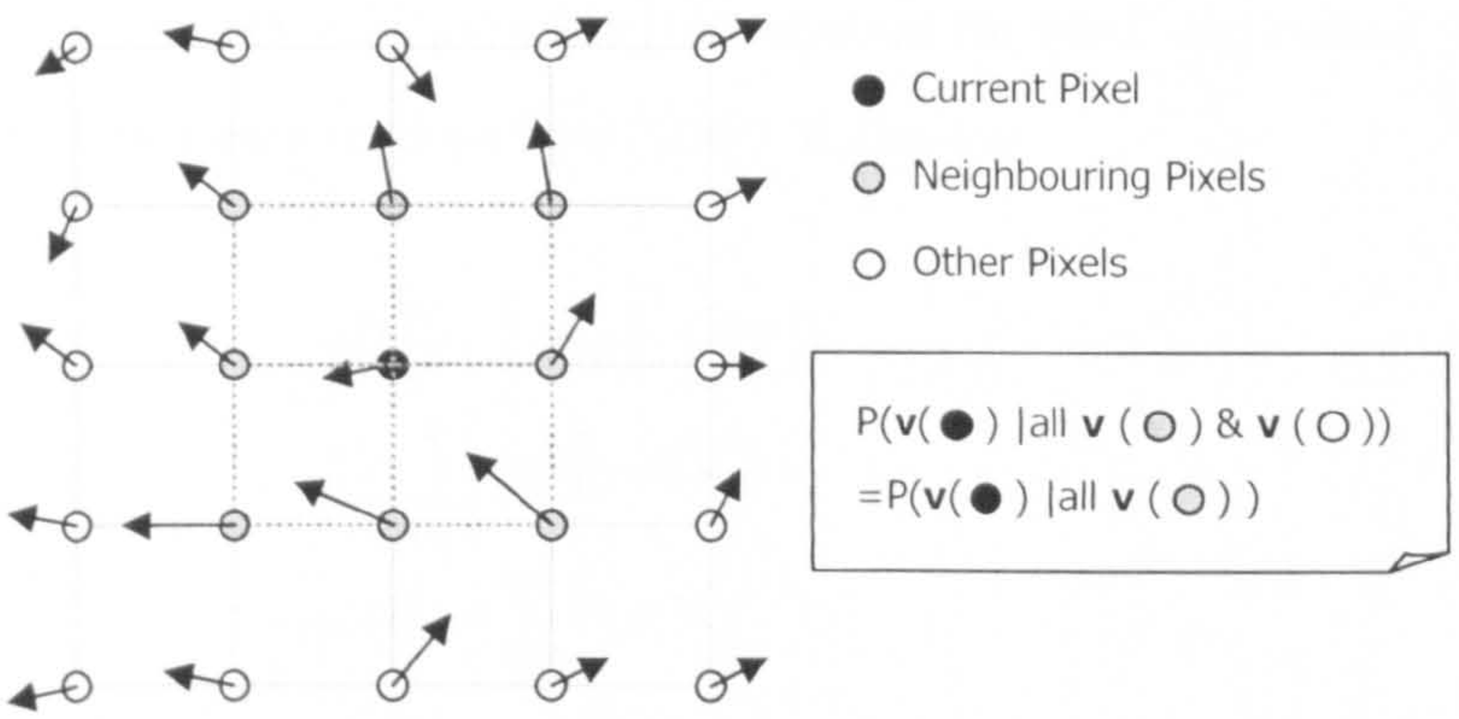


Figure 3.1 An illustration of motion vector field modelled as a Markov random field.
The conditional probability of current pixel given other pixels is fully determined by the conditional probability of current pixel given its neighbouring pixels.

Dropping the intrinsic t variable to always refer to the current frame, the conditional probability of current pixel $\mathbf{v}(\mathbf{p})$ is based on a neighbourhood system $\eta(\mathbf{p})$ such that:

$$\left.\begin{aligned} &\mathbf{p} \notin \eta(\mathbf{p}) \\ &\eta(\mathbf{p}) \in \mathbf{q} \Leftrightarrow \eta(\mathbf{q}) \in \mathbf{p} \end{aligned}\right\} \forall \mathbf{p}, \mathbf{q} \in \mathbf{\Lambda} \quad \text{Eq 3-21}$$

Common neighbourhood systems are the 4-neighbour system η_4 and 8-neighbour system η_8:

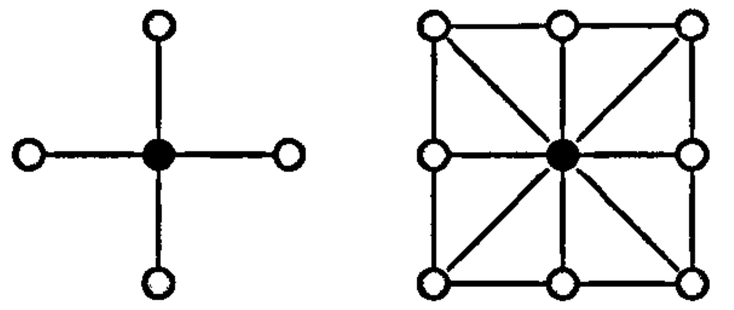


Figure 3.2 4-neighbour and 8-neighbour systems.

A Markov random field (in this case the motion vector field $\mathbf{V} = \{\mathbf{v}(\mathbf{p}): \mathbf{p} \in \Lambda\}$) is defined as:

$$p(\mathbf{v}(\mathbf{p}) \mid \{\mathbf{v}(\mathbf{q}): \mathbf{q} \in \Lambda / \{\mathbf{p}\}\}) = p(\mathbf{v}(\mathbf{p}) \mid \eta(\mathbf{p})) \qquad \text{Eq 3-22}$$

A Markov random field specified in terms of conditional probabilities is not of much use since the configuration probability, $p(\mathbf{V} = \{\mathbf{v}(\mathbf{p}): \mathbf{p} \in \Lambda\})$ is required for MAP estimation. This is circumvented by the equivalence of the Gibbs random field (GRF), defined as:

$$p(\mathbf{V}) = \frac{1}{Z}\exp\{-U(\mathbf{V})\} \qquad \text{Eq 3-23}$$

$$Z = \sum_{all\ \mathbf{V}} \exp\{-U(\mathbf{V})\}$$

$$U(\mathbf{V}) = \sum_{c \in C} V(c; \mathbf{V})$$

In Eq 3-24, Z is the partition function which normalizes $p(\mathbf{V})$ to make it a probability measure. C is the collection of cliques of the neighbourhood system, defined as the set of neighbouring pixels. For instance the clique of a 4-neighbour system can be a pixel, or a pair of neighbouring pixels; no 3 pixels can form a clique in 4-neighbour system. The term $V(c; \mathbf{V})$ evaluates into a real number representing the "energy" of the clique – a higher energy lowers the probability $p(\mathbf{V})$. We can now define quote and example of a prior probability of the motion vector field in terms of a GRF, using the difference between neighbouring motion vectors as clique energy and a 4-neighbour system:

$$p(\mathbf{V} \mid \mathbf{I}_{t-1}) = \frac{1}{Z}\exp\left(-\sum_{all\{\mathbf{p},\mathbf{q}\ \mathbf{p}\in\Lambda,\mathbf{q}\in\Lambda,\mathbf{p}\neq\mathbf{q}\}} \|\mathbf{v}(\mathbf{p}) - \mathbf{v}(\mathbf{q})\|\right) \qquad \text{Eq 3-24}$$

Finally motion estimation by Bayesian MAP methods is expressed as Eq 3-25 by substituting Eq 3-20 and Eq 3-24 into Eq 3-8.

$$\mathbf{V} = \arg\max_{all\ \mathbf{U}} p(\mathbf{U}\,|\,\mathbf{I}_t,\mathbf{I}_{t-1}) = \arg\max_{all\ \mathbf{U}} p(\mathbf{I}_t\,|\,\mathbf{U},\mathbf{I}_{t-1})p(\mathbf{U}\,|\,\mathbf{I}_{t-1}) \qquad \text{Eq 3-25}$$

$$= \arg\min_{all\ \mathbf{U}}\left\{\sum_{\mathbf{p}\in\Lambda}\frac{[I(\mathbf{p},t)-I(\mathbf{p}-\mathbf{v}(\mathbf{p}),t-1]^2}{2\sigma^2}+\sum_{\{(\mathbf{p},\mathbf{q})\in\Lambda^2,\mathbf{p}\neq\mathbf{q}\}}\|\mathbf{v}(\mathbf{p})-\mathbf{v}(\mathbf{q})\|\right\}$$

Finding the optimal configuration is a highly complex task; this has been, and still is a vast area of research. In summary, the various algorithms can be categorized into two main groups – simulated annealing and deterministic annealing.

Simulated annealing [Kir-83] involve a class of stochastic relaxation techniques called the Monte Carlo methods. The process starts with perturbing an initial configuration, and comparing the probability of the perturbed field with the original field, if the new probability is higher, the new field is accepted and the process repeats. If the new probability is lower, the perturbed field is not immediately discarded; it has a certain probability of being selected, which is dependent on a "temperature" value. The iteration continues for a pre-determined number of steps, after which the temperature value is reduce, thus decreasing the chances of accepting any less likely configurations. This whole process eventually reaches a stable condition with the final field converging to an optimal result. Two popular schemes of relaxation are the Metropolis algorithm and the Gibbs sampler [Gem-84].

In contrast to simulated annealing, deterministic annealing methods [Ros-98] do not accept less likely configurations. This makes deterministic annealing methods converge fast, but with an increasing risk of reaching a local minimum. Hence it is very important to have a good initial guess before the deterministic annealing process begins. Typical examples of deterministic annealing include Iterative Conditional Modes (ICM) [Bes-86], Mean Field Annealing (MFA) [Abd-92] [Zha-93] and Highest Confidence First (HCF) [Cho-90].

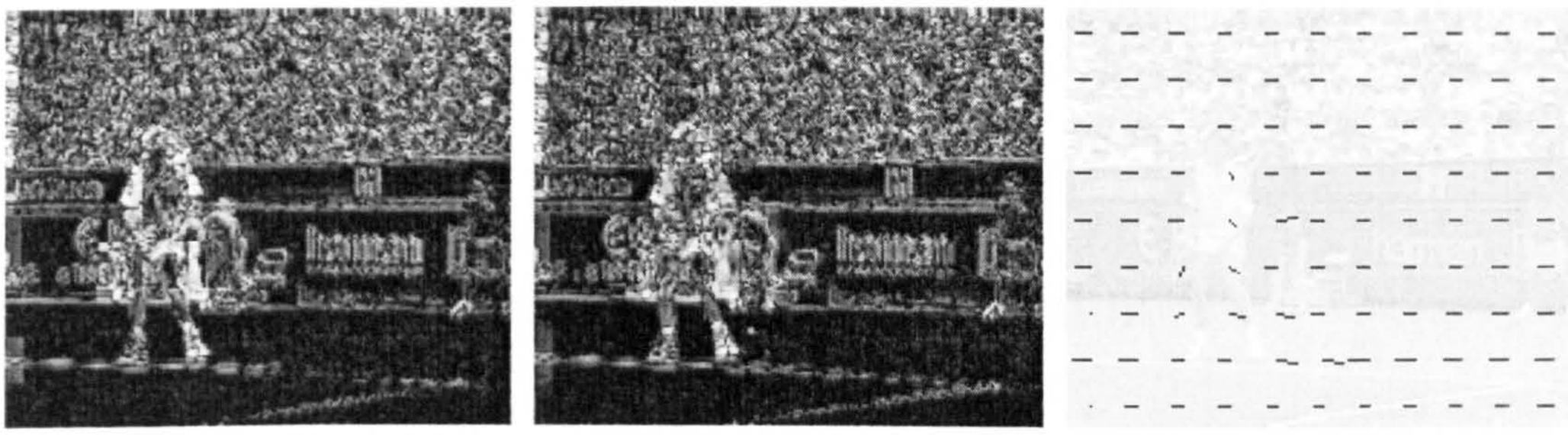
3.2.4 Region-Based Matching Methods

Region-based matching methods are purely heuristic approaches to motion estimation. A typical example is the block matching algorithm (BMA) used in all video coding standards. Other methods include phase correlation, motion modelling and quad-tree-based split and merge matching. This thesis focused on some novel approaches to BMA and aims at exploiting the speed of this method to segment video sequences into more natural regions, so video can be compressed at a higher level.

Some of the issues which will not be covered in detail but are worth mentioning in the section are the overlapped block motion estimation and the multiple reference frame based motion estimation.

BMA methods usually produce residual images which contain very sharp edges. This is partly due to block-boundaries as well as the motion failure in uncovered regions. These sharp edges carry a wide range of spatial frequencies which, when transformed, require a lot more bits to code. To mitigate this

effect, the overlapped block motion compensation (OBMC) [Kuo-98] [Lee-99] [Tao-97] is used (in H.263), which averages pixel values from the motion due to current block and the neighbouring block. As illustrated in Figure 3.3, the displaced frame difference with OBMC (b) has softer edges compared with traditional motion compensation (a). OBMC has been proved to improve coding efficiencies by removing sharp edges, thus reducing high frequency components. The framework laid out in this thesis can utilise OBMC directly without much difficulty and will be left as future work to be done beyond the scope of this thesis.



(a) MSE = 276 (b) MSE = 221 (c)

Figure 3.3 An illustration of advantage of using overlapped block motion compensation (OBMC) with BMA: (a) shows a DFD from normal BMA; (b) shows a DFD from BMA with OBMC and (c) shows the motion vector field. Comparison of (a) and (b) reveals that DFD in (b) has less sharper edge. A higher mean-squared-errors (MSE) in (a) reinforces the point.

As an object moves across a static background, different regions are uncovered and occluded. Uncovered background regions in the current frame may not find a match with the immediate previous frames, as illustrated in Figure 3.4. By using multiple reference frames, the matching process can be carried out across a multitude of past frames. This has been shown to be effective in scenes with multiple moving objects of small areas. However, the process is very memory-intensive and introduces significant computational load to the video encoder. As this method is a logical extension from single reference frame, the incorporation of multiple reference frames into BMA will not be examined in this thesis.

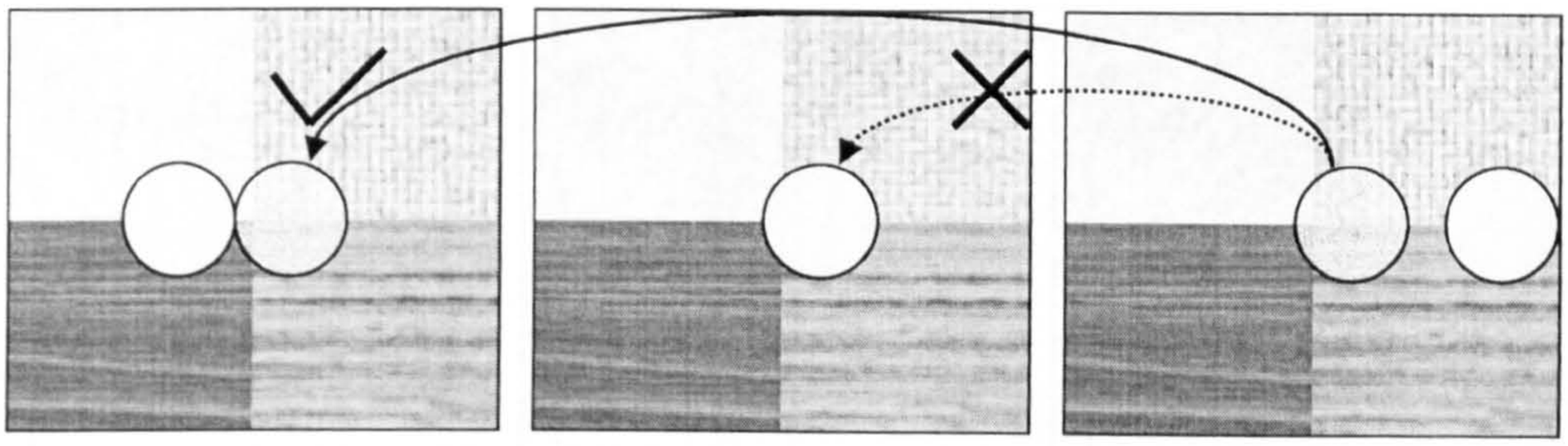
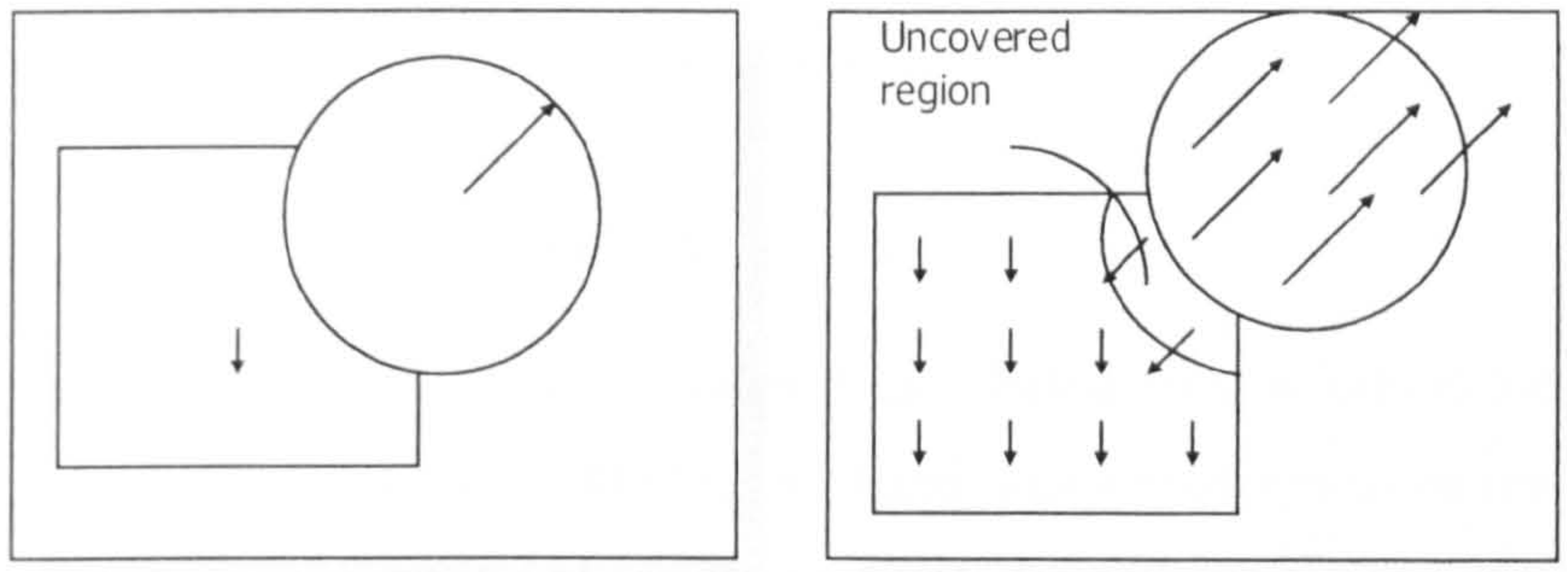


Figure 3.4 Illustration of solving occlusion problem with multiple reference frames.

3.3 Some Considerations in Motion Estimation

3.3.1 Occlusion Problem

Regardless of the methods used to solve the motion estimation problem, it is usually assumed that the motion at the particular pixel or region does exist. This is not necessarily the case in a previously occluded background region revealing itself as a moving foreground object moves. Motion vectors found in these uncovered regions are meaningless, at best. From the coding point of view such motion vectors simply points to the region in the reference frame which produces the best predictor; for motion segmentation and global motion estimation, such spurious deviation from true motion tend to sway estimates away from the true solution. As such, an attempt has to be made in the next-generation coding techniques to remove them from the estimation process. The use multiple reference frames as describe in the previous section is practical approach to alleviate this problem, provided there is enough memory and processing resources.



Frame t-1 and motion of two objects

Frame t and motion vector field of two objects and an uncovered region

Figure 3.5 Illustration of motion vectors in uncovered region.

3.3.2 Aperture Problem

A complementary problem to occlusion is the aperture problem. In the former, there is no meaningful solution to motion estimation, whereas in the latter, there is more than one solution. Due to the lack of texture and also due to the fact that motion along the direction normal to the intensity gradient cannot be estimated, real motion may not be found, as shown in Figure 3.6. The white square foreground is moving in the north-east direction, a moving corner contained in block B allows motion to be estimated accurately. On the other hand, block A has only a horizontal edge, which makes the horizontal component of the motion impossible to be estimated; similarly the vertical component of the motion in block C cannot be estimated without ambiguity.

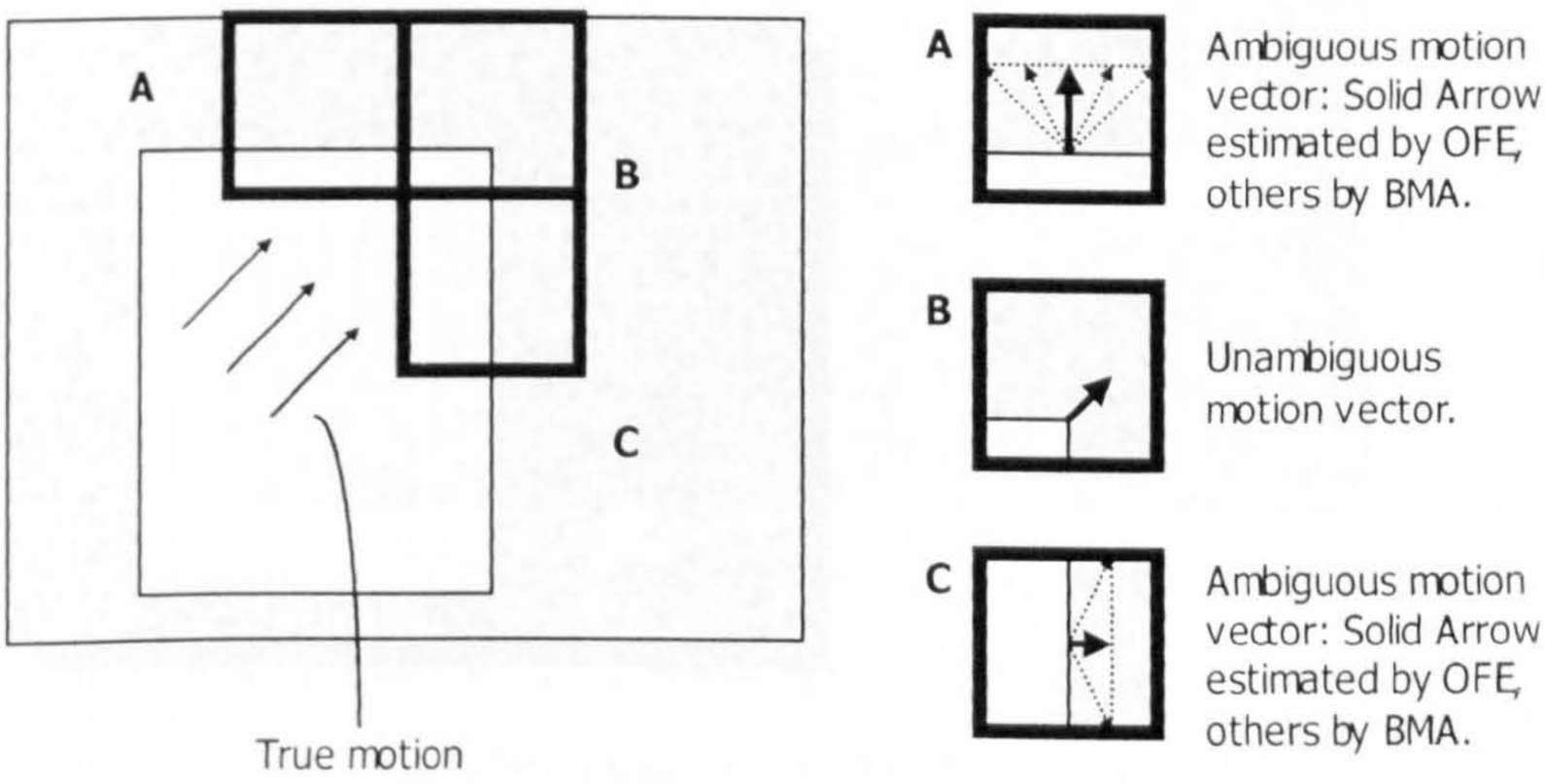


Figure 3.6 Illustration of aperture problem with motion estimation.

The wrong choice of motion vector may not be a serious problem insofar as the estimated vector results in the best match instead of the true match which may be not be the optimal one due to noise corruption

or a change in local intensity (an example would be an object moving into the shadow of another). However, such deviants from the 'ground truth' often lead to discontinuities in the motion vector field, requiring more bits to code in a predictive coding system.

3.3.3 Varying Block-size for BMA

A classic problem pertaining to BMA and other region-based motion estimation algorithms is the choice of block size or area of a region. On the one hand, block sizes are to be made large enough so that an accurate vector can be estimated in the presence of small 'noisy' regions; larger block sizes usually include more texture so that the vector found by block matching is more indicative of the motion of the region as a whole. On the other hand, choosing too large a size increases the risks of having the multiple moving objects within a single block. This is frequently referred to as the general aperture problem, where the choice of windows sizes involves getting enough texture for an accurate estimate whilst making sure the size is not too large to include multiple moving objects.

Figure 3.7 shows a typical segmentation of a BMA using varying block size to compromise between having enough texture to have a robust estimate and containing multiple objects. This method has also been used in various rate-distortion optimization schemes to jointly optimize the amount of residue and the amount of side information needed for partitioning and motion vectors.

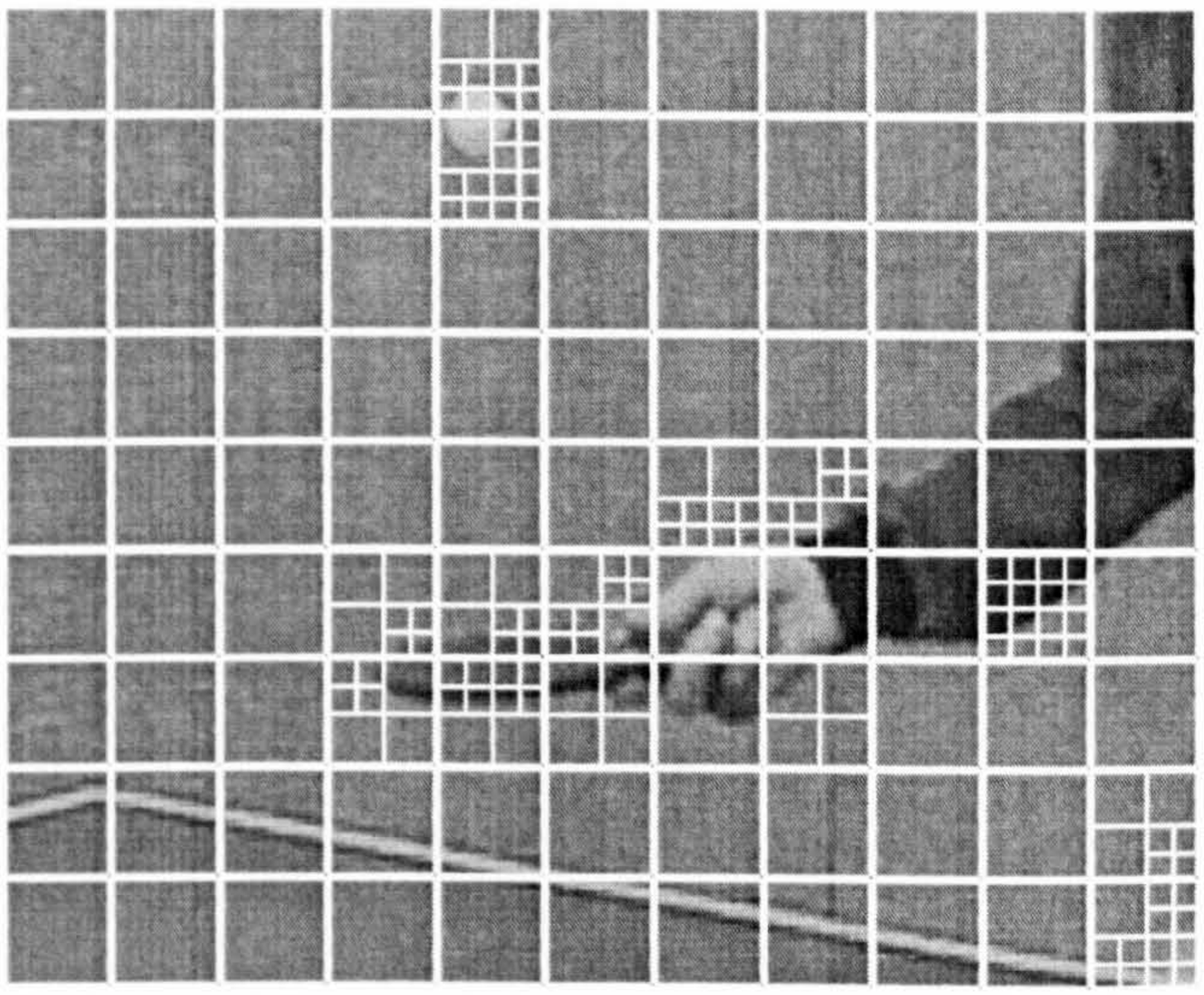


Figure 3.7 Typical quad-tree segmentation resulting from varying block size according to compromising between robustness and multiple objects.

3.4 Summary

This chapter begins with the basic principle behind motion estimation. The problem can be expressed as the solution to the optical flow equation, the matching of regions according to certain cost function or similarity measure, or the maximization of the a posteriori probability that a motion vector field occurs as a result of the two frames subject to certain prior probability constraints. Based on these principles, a few LME methods are reviewed. These include methods using OFE, the pel-recursive methods, Bayesian methods and the region-based matching methods.

The first three categories have traditionally been confined within academic fields; in practical situations these solutions require large complex systems and usually entail off-line processing. Real-time LME like video compression and motion detection uses region-based methods almost exclusively. In particular, the block-based matching (BMA) approach is used in all known video compression standards because it is computationally light-weight and conceptually simple. However, BMA is sometimes plagued with problems like the generalized aperture problem. In the next chapter, we introduce a few novel modifications to the traditional BMA method which brings about improvements in computation times, estimation accuracies and precisions. The next chapter also shows how elements from the first three LME methods can be incorporate into BMA in real-time to evaluate a motion vector field which is either closer to the 'ground truth', or requires less bit to code without introducing extra residues.

Chapter 4:

Novel Approaches to BMA

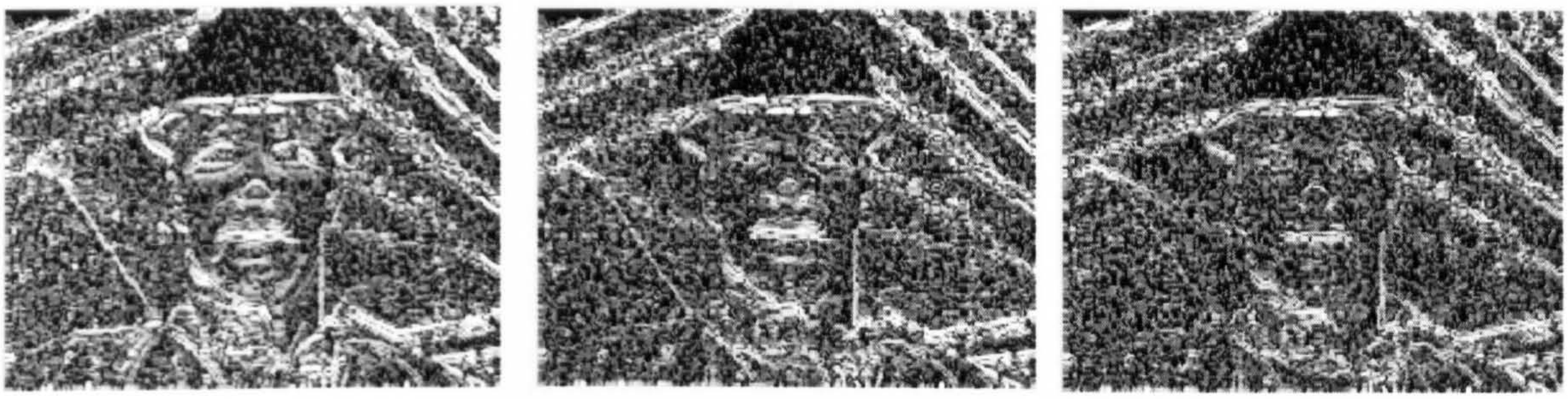
In the previous chapter, various local motion estimation methods were discussed. Amongst the methods discussed, the Block Matching Algorithm (BMA) is touted as the basic motion estimation algorithm of choice in video coding standards, and is also the most plausible candidate for real-time picture sequence analysis in machine vision. Its computational simplicity makes it ideal for real-time video compression and analysis. Research in the past has focused on fast search algorithms to find sub-optimum solutions to the minimization problem. As hardware and processor speed increases, algorithms performing full search has been made possible.

However, memory requirements and bandwidth has become the new bottleneck for most processing systems. As opposed to other local motion estimation methods, motion vectors resulting from BMA suffers from the lack of sub-pixel resolution; in order to achieve sub-pixel resolution for the vectors, reference pictures are usually interpolated to the desired sub-pixel level and motion is then estimated based on these interpolated pictures, Sub-pixel estimation in BMA are known to bring about substantial compression improvements, but interpolation of reference frames usually entails large memory storages and substantial processor time is spent on memory access. Access to these large interpolated frames increases memory bandwidth, which tends to stall most processing systems as pipelines are broken. The next section introduces a sub-pixel estimation of BMA without interpolation. This is done by modelling the distortion space around the integer-pixel-based minimum point.

Subsequent sections of this chapter introduce a new way of calculating a motion vector field. As opposed to the traditional raster scan method, the new algorithm uses a novel queue-based approach which processes 'reliable' blocks first. Subsequent less reliable blocks have an additional smoothness constraint tagged to their minimization objective. This new algorithm involves introducing a new reliability measure to a BMA, adding a smoothness constraint term into the minimization problem similar to the Horn and Schunck formalization, and a queue-based approach similar to HCF [Cho-90] of the deterministic annealing method. It introduces a better smoothness constraint which can either be used to produce a more natural field for motion segmentation, or to reduce the entropy of the 'smoothed' vector field.

4.1 Interpolation-free Sub-pixel Estimation for BMA

A popular method of motion estimation in video coding system, Block Matching Algorithms (BMA) offers speed and robustness where other pixel-based methods cannot. One short-fall of BMA is the fact that with the original reference frame, motion vectors can only have integer resolution. Motion estimation at sub-pixel resolution provides more efficiency for BMA, thus reducing the residual displaced frame difference. As shown in Figure 4.1, the DFD energy reduces gradually as motion estimation (BMA at 8x8 blocks) is carried out at increasing sub-pixel resolution.



(a) Full-pixel. (PSNR=35.4 dB) (b) 1/2-pixel (PSNR=37.8 dB) (c) 1/4-pixel (PSNR=39.0 dB)

(d) 1/8-pixel (PSNR=39.5 dB) (e) 1/16-pixel (PSNR=39.6 dB)

Figure 4.1 Illustration of the benefits of sub-pixel motion estimation. There is a gradual reduction in the high-energy pixels in the displaced frame difference (DFD) as sub-pixel resolution increases, as quantified by the PSNR.

Sub-pixel resolution can be found via interpolating fractional pixel values. This is used in major video coding standards [ITU-93] [ITU-98] [MPE-93] [MPE-95], where pixels are represented at 1/2-pixel resolution for the luminance component and 1/4-pixel resolution for the chrominance components. H.264 [JVT-02] also allows 1/4 –pixel motion estimation (1/8 resolution was proposed at one time, but was subsequently dropped due to its associated computational load and lack of substantial coding improvements over that achieved by 1/4-pixel resolution). Motion estimation in such cases requires a pre-processing step to interpolate the reference frame. Interpolated frames require memory to store them, Table 4.1 shows the amount of memory required to store these interpolated frames. In embedded systems with VLSI and DSP implementation, such quantities of memory may not be justifiable; even if

such memory can be added as secondary storage area, which are usually much slower, data transfer between the memory and the VLSI/DSP devices can greatly impact performance. Hence sub-pixel refinement with interpolated reference frames is usually not used in real-time mobile solutions. In applications like motion segmentation and global motion estimation where an optimal solution is to be obtained iteratively, motion vectors at sub-pixel resolution are very crucial to ensure rapid convergence and convergence to the true minimum.

Table 4.1 Memory requirements in (bytes) for storing reference pictures.

	Original Picture	**½-pel interpolated Picture**	**¼-pel interpolated Picture**
QCIF	38 kB	152 kB	608 kB
CIF	152 kB	608 kB	2.43 MB

To date, there has not been systematic research on methods for estimating sub-pixel values without prior interpolation of the reference frame. An exception is motion estimation in the DCT domain [Koc-96] [Koc-98], where sub-pixel processing is intrinsic in the phase estimation in the frequency domain. However adopting such algorithms requires a complete change of the video coding structure common to all standards. There have also been investigations into how sub-pixel locations can be searched efficiently [Pan-02], but these are done by actually interpolating the frame itself during every search points. The following sections introduce a sub-pixel motion estimation method within the common video coding structure depicted in Figure 2.13. In practice, the results of sub-pixel motion estimation without using the interpolated reference frame would never be better than those actually the using interpolated frames (referred from here onwards as the 'FullSub' method). What we aim for is to reduce processing time while achieving results as close to that achievable by the FullSub method as possible. The scheme will be very useful to applications where interpolation is not feasible due to the real-time constraint imposed by either physical memory access and/or processor power. A similar method can be found in [Dan-03] which uses one of the models we proposed. As will be made clear later the model proposed by Dante and Mike is straight-forward but does not provide a better estimate to the sub-pixel motion vector in PSNR sense. The subsequent sections propose a better model which is computationally less intensive and provides a better sub-pixel estimation of the motion vectors.

4.1.1 Model Description

This thesis proposes an interpolation-free sub-pixel motion estimation algorithm. Integer-based BMA is used to estimate the motion vector of each block at integer-pixel resolution. Then the SAD's of the candidate motion vector and its surrounding 8 neighbours in the motion vector space are assumed to be parabolically distributed along the x- and y-axes, as described in Eq 4-1 [Giu-99]:

$$S(x,y) = Ax^2 + By^2 + Cxy + Dx + Ey + F \qquad \text{Eq 4-1}$$

Where $S(x,y)$ is the SAD at co-ordinates (x,y). The x- and y- ordinates are centred at the motion vector at integer-pixel resolution and vary between –1 and 1. A, B, C, D, E and F are the parameters whose values are to be estimated for each block.

To arrive at Eq 4-1, we start with the second order Taylor's expansion of $S(x,y)$ around the minimum at integer-pixel resolution (x_0, y_0):

$$S(x,y) = S(x_0,y_0) + \begin{bmatrix} \frac{\partial S}{\partial x} & \frac{\partial S}{\partial y} \end{bmatrix} \begin{bmatrix} x-x_0 \\ y-y_0 \end{bmatrix} + \begin{bmatrix} x-x_0 \\ y-y_0 \end{bmatrix}^T \begin{bmatrix} \frac{\partial^2 S}{\partial x^2} & \frac{\partial^2 S}{\partial x \partial y} \\ \frac{\partial^2 S}{\partial x \partial y} & \frac{\partial^2 S}{\partial y^2} \end{bmatrix} \begin{bmatrix} x-x_0 \\ y-y_0 \end{bmatrix} \qquad \text{Eq 4-2}$$

Simplifying Eq 4-2 and replacing $\begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} x-x_0 \\ y-y_0 \end{bmatrix}$:

$$S(x,y) = S(x_0,y_0) + \frac{\partial S}{\partial x}(\Delta x) + \frac{\partial S}{\partial y}(\Delta y) + \frac{\partial^2 S}{\partial x^2}(\Delta x)^2 + \frac{\partial^2 S}{\partial y^2}(\Delta y)^2 + 2\frac{\partial^2 S}{\partial x \partial y}(\Delta x)(\Delta y) \qquad \text{Eq 4-3}$$

Changing the reference grid to centre at (x_0, y_0), $(\Delta x, \Delta y)$ becomes (x, y) and Eq 4-3 translates into Eq 4-1.

To obtain the minimum point of the model in the $(-1,+1)\times(-1,+1)$ area, we can either:

1. Evaluate the extremum location via the partial derivatives of Eq 4-1; or
2. Evaluate each and every candidate points at the desired sub-pixel resolutions and find the minimum.

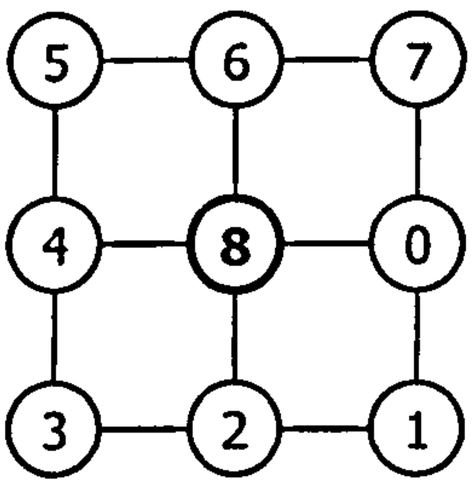
We chose the second approach because:

1. The first approach may result in non-minimum point or a minimum point beyond the ±1 window;
2. Direct evaluation of candidate points yields results faster by avoiding division operations.

Hence with the model parameters found for each block, the SAD of eight half-pixel candidates (±½,0), (±½,±½), (0, ±½) are evaluated according to Eq 4-1 for half pixel accuracy in H.263 and an additional 40 candidates at quarter-pixel resolution for H.264.

To model the SAD distribution around the integer minimum point, nine points can be used – the point at which the minimum SAD occurs at integer pixel (0,0), and its neighbouring eight points (1,0), (1,1), (0,1) (-1,1), (-1,0) (-1,-1), (0,-1) and (1,-1). The points are indexed as shown in the Figure 1. Point 8 is

the minimum point in the integer-pixel resolution block matching cost function of the current block. Even-numbered points are called the near-neighbours of point 8 and the remaining ones its far-neighbours.



Pixel co-ordinates with respect to centre pixel:

Pixel 8: (0,0)	Pixel 2: (0,1)	Pixel 5: (-1,-1)
Pixel 0: (1,0)	Pixel 3: (-1,1)	Pixel 6: (0,-1)
Pixel 1: (1,1)	Pixel 4: (-1,0)	Pixel 7: (-1,1)

Centre pixel: Pixel 8
Near neighbours: Pixels 0, 2, 4, 6
Far neighbours: Pixels 1, 3, 5, 7

Figure 4.2 Illustration of neighbouring pixel indices.

Eq 4-4 shows the SAD values and their respective co-ordinates of the nine observation points. With a system of 6 variables and 9 equations, we explore the following three models that can be adopted to evaluate the parameters:

1. Near-neighbours model (NNM)
2. Complete-system model (CSM)
3. Over-complete-system model (OSM)

$$
\begin{aligned}
S_0 &= S(1,0) = A + D + F \\
S_1 &= S(1,1) = A + B + C + D + E + F \\
S_2 &= S(0,1) = B + E + F \\
S_3 &= S(-1,1) = A + B - C - D + E + F \\
S_4 &= S(-1,0) = A - D + F \\
S_5 &= S(-1,-1) = A + B + C - D - E + F \\
S_6 &= S(0,-1) = B - E + F \\
S_7 &= S(1,-1) = A + B - C + D - E + F \\
S_8 &= S(0,0) = F
\end{aligned}
\qquad \text{Eq 4-4}
$$

4.1.1.1 Near-neighbours Model

In this model, C is set to zero and the remaining parameters are solved using S_8 and its 4 near-neighbours, as illustrated in Eq 4-5.

This model assumes S varies with x and y independently. It is the simplest model to implement and is ideal in cases where the minimum is less well defined and the block does not contain a strongly directional pattern. Moreover, the true minimum point is guaranteed to occur within the fractional-pixel

range. However this minimum point is constrained within the $[-0.5,0.5]$ range, which makes it unsuitable to locate true minimum points which lie beyond this $[-0.5,0.5]$ window.

$$A=-S_8+\frac{1}{2}(S_0+S_4)\ ;\ B=-S_8+\frac{1}{2}(S_2+S_6)\ ;\ C=0$$
$$D=\frac{1}{2}(S_0-S_4)\ ;\ E=\frac{1}{2}(S_2-S_6)\ ;\ F=S_8 \qquad \text{Eq 4-5}$$

4.1.1.2 Complete-System Model

In this model, one of the far-neighbours with the lowest SAD value is added to the NNM solution set to form a complete system of equations. The model parameters A, B, D, E and F have the same value as those found in Eq 4-5; the remaining parameter C is chosen from the set of $\{C_1,C_3,C_5,C_7\}$ where C_k is the value of C found with the complete system of equation using points 0, 2, 4, 6, 8 and k. Then the value of C is chosen using Eq 4-6, where $\hat{S}_{ki}$ is the estimate of S_i using Eq 4-4 with parameter set $\{A,B,C_k,D,E,F\}$. This model determines which of the far-neighbours best fits the model and ignores the other 3, thus removing the effects of outliers. Its complexity is similar to that of the NNM and also guarantees the existence of a minimum point.

$$C=\arg\min_{k=1,3,5,7}\sum_{i=1,3,5,7}\left|S_i-\hat{S}_{ki}\right| \qquad \text{Eq 4-6}$$

4.1.1.3 Over-complete-System Model

The OSM model uses all 9 points to form an over-complete equation system depicted in Eq 4-7. To solve the system of equations, the least squares method of pseudo-inverse is used, giving the solution in Eq 4-8. This method is most complex and tends to produce a 'flatter' model, which makes the location of the minimum point less defined. As one or more observation points may not necessary lie on the modelled surface, this can produce a saddle-point instead of a minimum point. An added disadvantage is the excessive use of multiplication and division in the matrix operations, which may not be feasible in real-time applications.

$$\begin{bmatrix} S_0 \\ S_1 \\ S_2 \\ S_3 \\ S_4 \\ S_5 \\ S_6 \\ S_7 \\ S_8 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & -1 & -1 & 1 & 1 \\ 1 & 0 & 0 & -1 & 0 & 1 \\ 1 & 1 & 1 & -1 & -1 & 1 \\ 0 & 1 & 0 & 0 & -1 & 1 \\ 1 & 1 & -1 & 1 & -1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} A \\ B \\ C \\ D \\ E \\ F \end{bmatrix} \quad \text{Eq 4-7}$$

$$\begin{bmatrix} A \\ B \\ C \\ D \\ E \\ F \end{bmatrix} = \frac{1}{36} \begin{bmatrix} -6 & 6 & -12 & 6 & 6 & 6 & -12 & 6 & -12 \\ -12 & 6 & 6 & 6 & -12 & 6 & 6 & 6 & -12 \\ 0 & -9 & 0 & -9 & 0 & 9 & 0 & -9 & 0 \\ 6 & 6 & 0 & -6 & -6 & -6 & 0 & 6 & 0 \\ 0 & 6 & 6 & 6 & 0 & -6 & -6 & -6 & 0 \\ 8 & -4 & 8 & -4 & 8 & -4 & 8 & -4 & 20 \end{bmatrix} \begin{bmatrix} S_0 \\ S_1 \\ S_2 \\ S_3 \\ S_4 \\ S_5 \\ S_6 \\ S_7 \\ S_8 \end{bmatrix} \quad \text{Eq 4-8}$$

4.1.2 Comparison of the Three Sub-pixel Models

To illustrate the actual distribution of the SAD values obtained by actual interpolated reference frame and that estimated by the three models, frame 200 of FOREMAN.QCIF is motion-estimated with frame 197 as a reference frame. 5 sample blocks from Figure 4.3 are extracted and their SAD-maps shown in Figure 4.4. To better show the SAD-distribution, estimations are done at 1/8-pixel resolution.

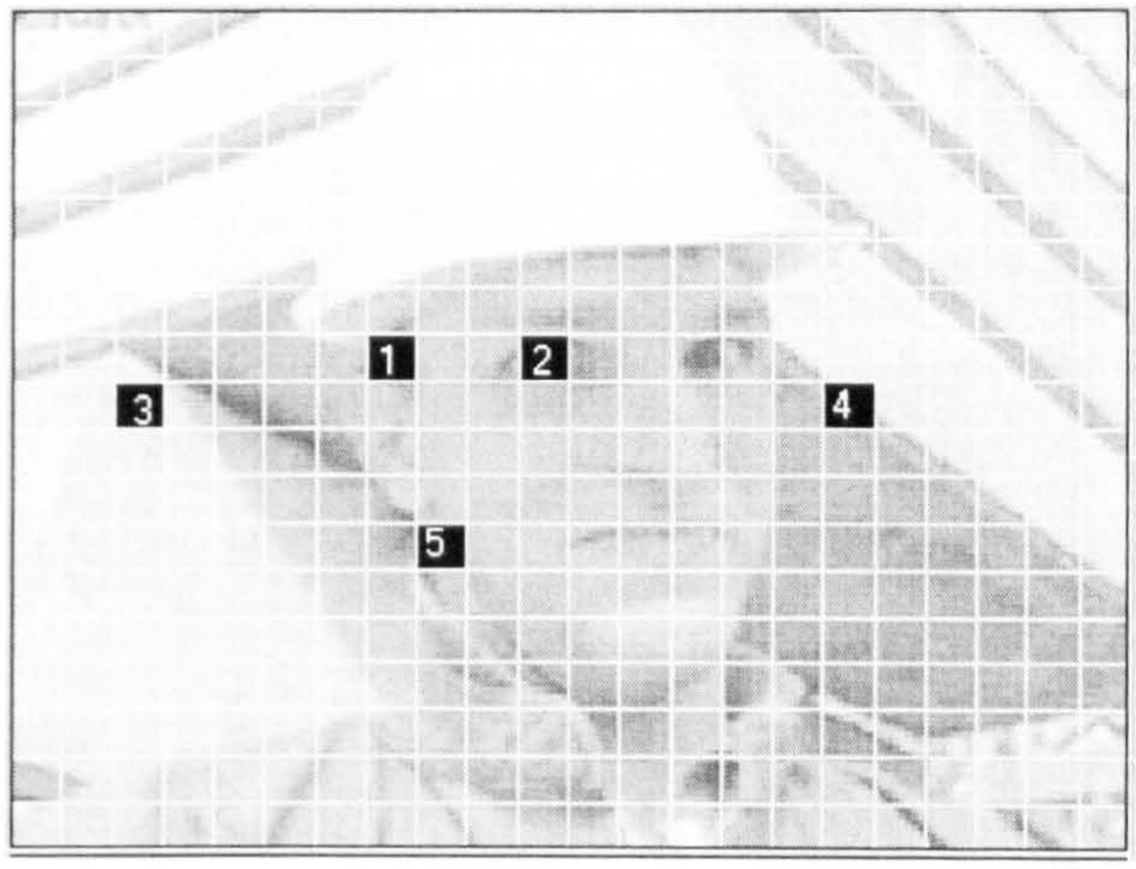


Figure 4.3 Frame 200 of FOREMAN.QCIF and 5 numbered blocks used to illustrate the sub-pixel SAD distribution around the candidate integer-pixel motion vector

The NNM's inability to model an asymmetric SAD distribution is evident from Figure 4.4. Furthermore, the possible solutions to Eq 4-5 are restricted to the [-0.5, 0.5] range; hence NNM tends to under-estimate the absolute value of the sub-pixel minimum point. CSM and OSM provide a better SAD model and a more accurate estimate of the minimum point. The OSM output tends to produce a 'flatter' surface due to the averaging effect of the over-complete system. It is also more prone to outlier effects that stray the result away from the actual minimum (blocks 3, 4). Under extreme cases the model produces a saddle point instead of a minimum point (block 4). CSM, on the other hand, sorts out the set of neighbours that are most likely to produce a better estimate and reject the others, which are deemed outliers. This process actually produces much better results, as will be shown in the simulation results in the following section.

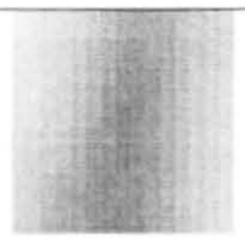
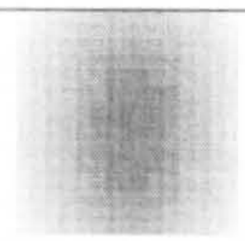
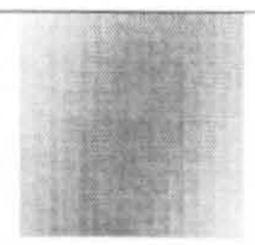
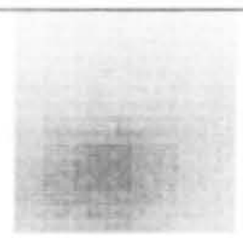
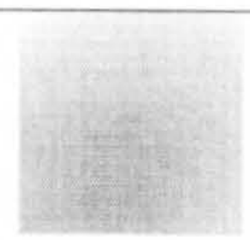
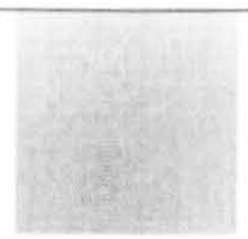
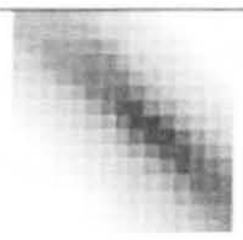
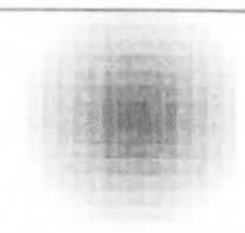
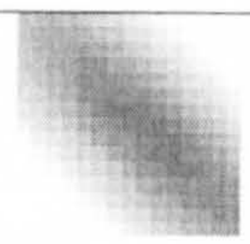
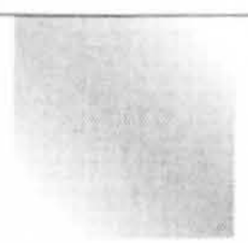
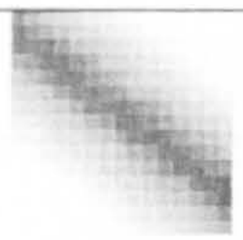
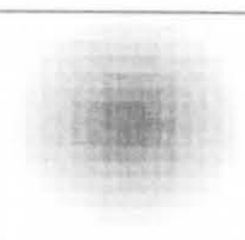
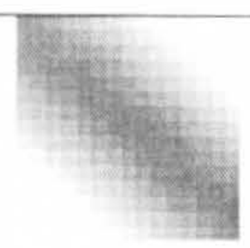
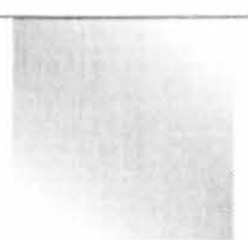
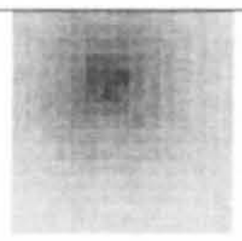
Blk	Sub-pixel maps at 1/8-pixel resolutions with location of the minimum point (values in 1/8-pixel units)			
	FullSub	NNM	CSM	OSM
1	 [0,1]	 [0,0]	 [0,1]	 [0,6]
2	 [-1,3]	 [0,1]	 [-1,3]	 [-1,3]
3	 [2,1]	 [1,0]	 [2,1]	 [6,6]
4	 [2,1]	 [0,0]	 [1,1]	 [-7,-7]
5	 [-1,-3]	 [0,-1]	 [-1,-3]	 [-1,-2]

Figure 4.4 The 1/8-pixel SAD distribution around the integer-resolution motion vectors of the 5 blocks. Second column is the SAD map found from the actual interpolated reference frame; The 3 numbers below each map denotes the horizontal and vertical components of the fractional motion vector at 1/8-pixel units

4.1.3 Simulation Results

This section compares the performance of the 3 models. The simulations are carried out on 2 types of sequences. The QCIF sequences at 10 frames per second (fps) are typical of real-time hand-held devices which will be most likely to benefit from the proposed algorithm due to power and size constraints; the CIF sequences at 30 fps are popularly used within the video streaming community for wireless channels. 8 standard test sequences of varying complexities are used. Prior to all sub-pixel refinement algorithms, a full-search BMA is carried out with a search window of 15 pixels to locate the

motion vector at integer-pixel resolution. In order to see the effect of block size of the BMA has on the effectiveness of the algorithms, simulations are carried out with 3 sizes: 16x16 blocks (basic macroblock size of all prevailing coding standards), 8x8 blocks (used in both H.264 and the advanced version of H.263) and 4x4 blocks (the smallest block size offered in the H.264 standards). To be in line with the new H.264 standard, sub-pixel estimation is done at ¼-pixel resolution.

Sub-pixel values in FullSub are obtained from bilinear interpolation of neighbouring integer-pixel values. Various interpolation filters have been evaluated which use longer filter length [Tri-01], but to minimize computational load, a simple bilinear interpolation is used (Figure 4.5).

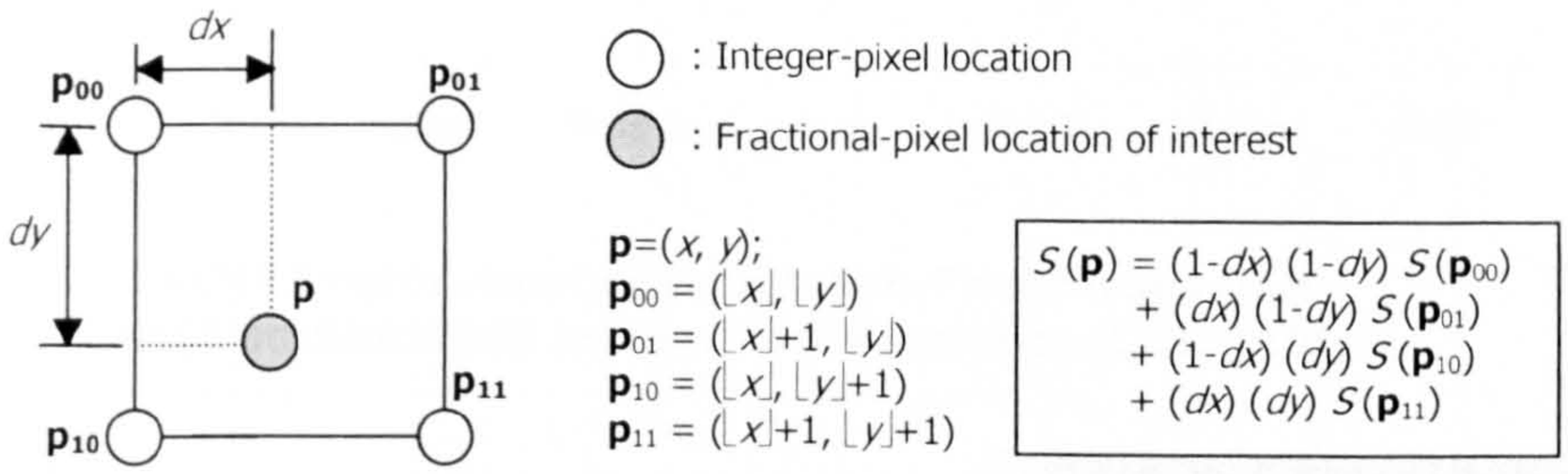
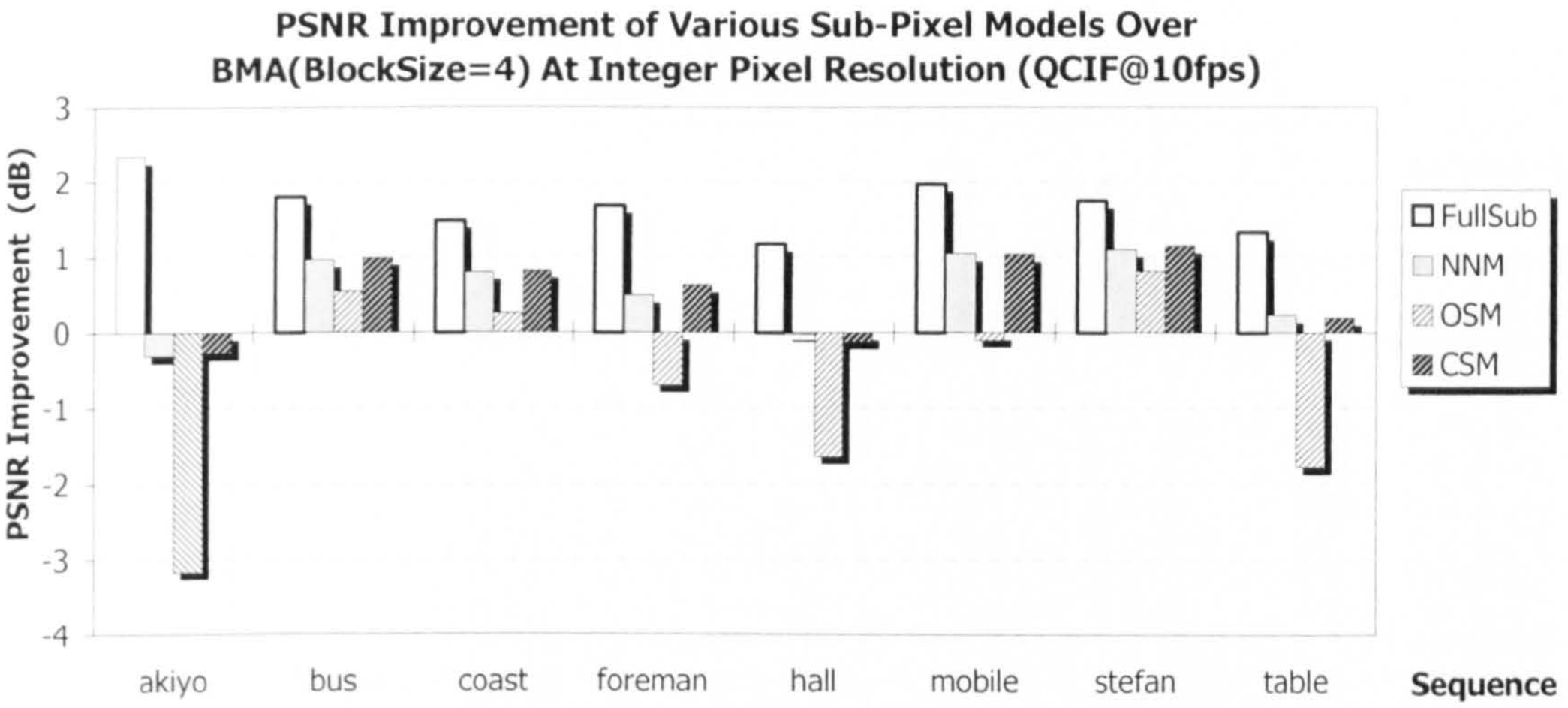
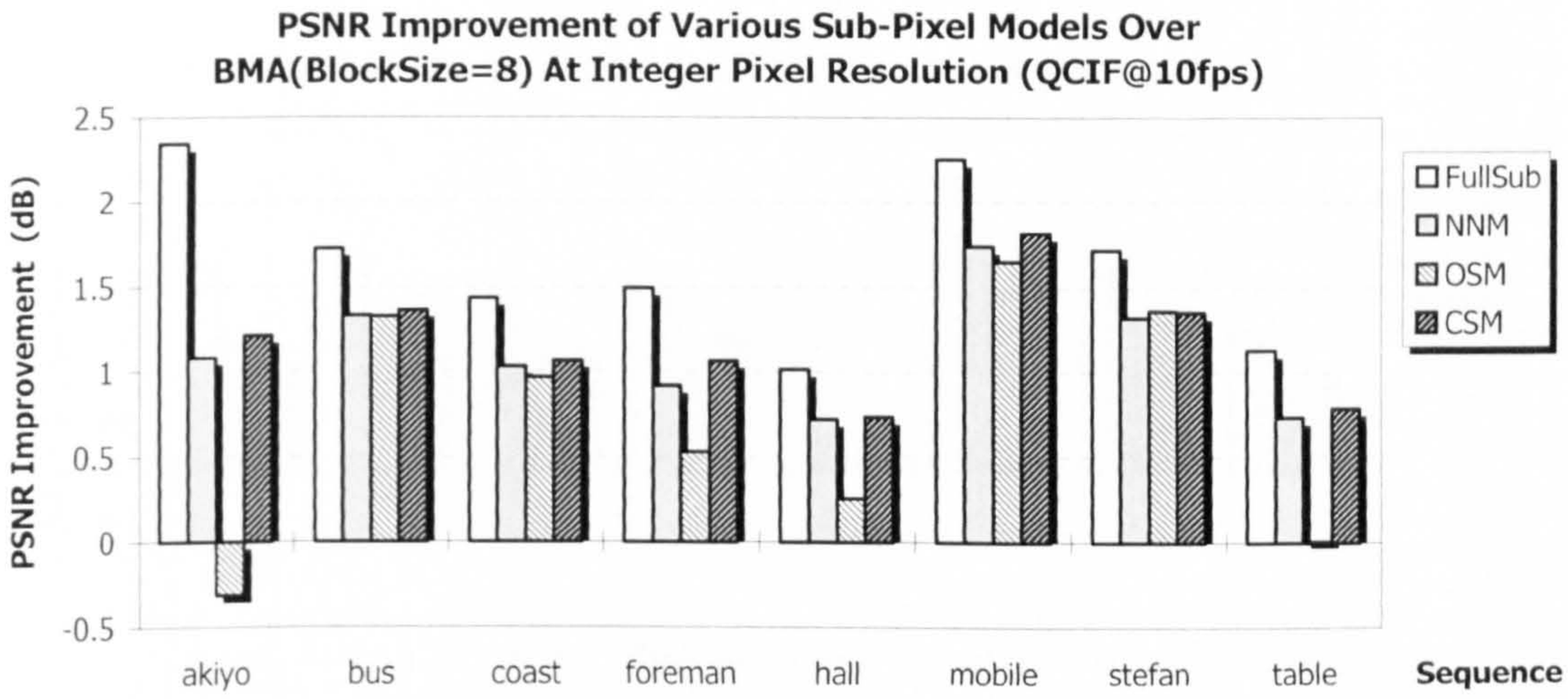


Figure 4.5 Illustration of bilinear interpolation of pixel **p** (x, y) from its 4 neighbouring integer-pixels. The fractional values of dx and dy are measurement from the top-left neighbour, and the $\lfloor n \rfloor$ operation returns the largest integer less then n.

Figure 4.6 and Figure 4.7 show the performances of the three sub-pixel models for QCIF@10fps and CIF@30fps sequences respectively. Across all sequences, there is a strong indication that all sub-pixel models work fairly with 16x16 blocks than with 4x4 blocks. The main reason for this is because the SAD distribution is smoother in larger block size and hence the quadratic model offers a better fit statistically. There is also a strong indication that the over-complete system model (OSM) is the worst-performing of the three models. In some instances, the model even fairs worse than the integer-resolution vectors. The best performing model is the complete system model (CSM) where 5 neighbouring points are selected to form a complete equation system. This model works better as it removes the possibilities of noise, which tends to corrupt the results obtained from OSM algorithm. NNM, on the other hand, stands between the other two models. It is not as sensitive to noise as the OSM, but unlike the CSM, it cannot model asymmetric distribution.

In conclusion, from our three algorithms, CSM offers the best estimate of the sub-pixel motion vectors in terms of PSNR of the predicted frame. The model provides approximately 50% to 75% of the improvement achievable by interpolated reference frame for block sizes of 8x8 and 16x16. Sub-pixel estimations do not appear to work as well for 4x4 blocks.





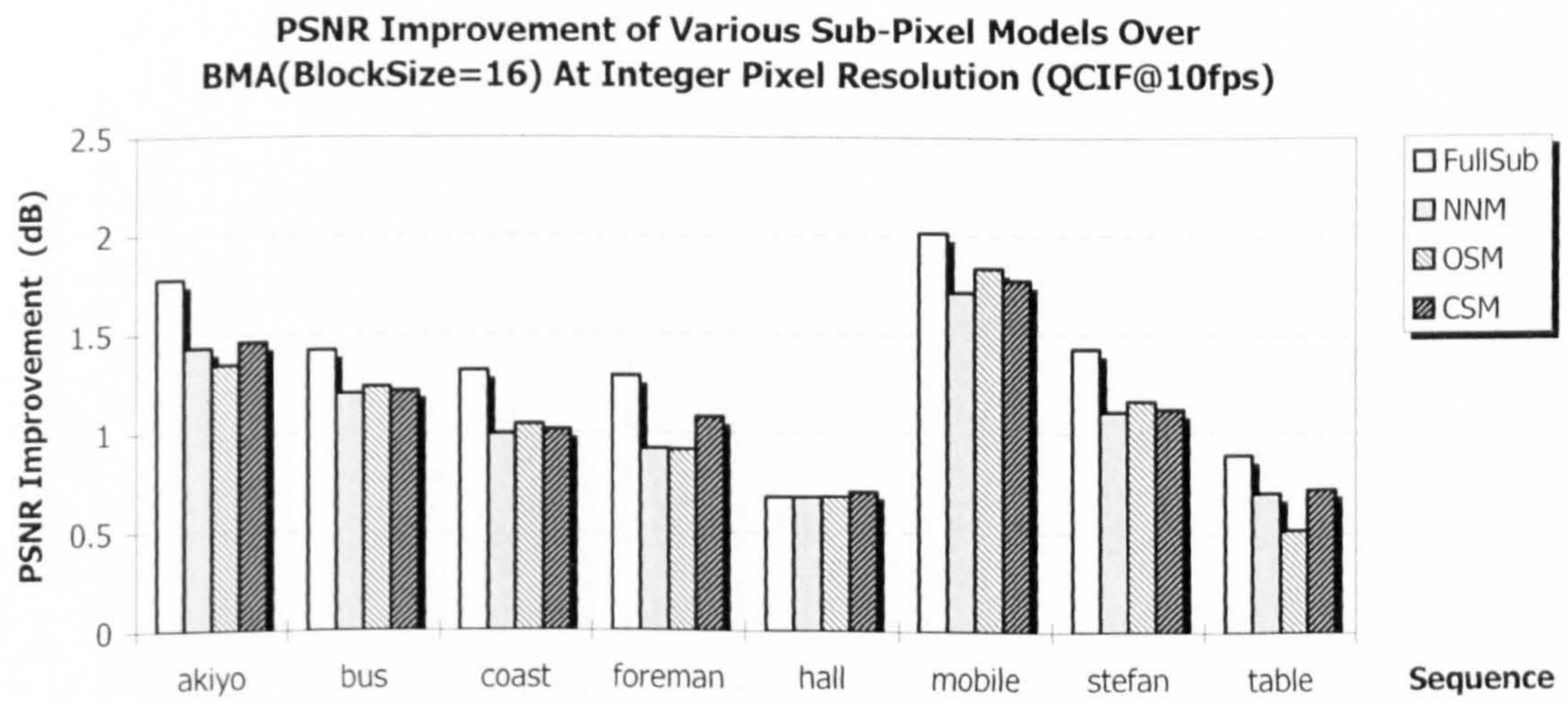


Figure 4.6 Bar charts showing the improvements of various sub-pixel model over integer-based BMA. Sequences are QCIF at 10 fps using: (top) 4x4 blocks (middle) 8x8 blocks and (bottom) 16x16 blocks.

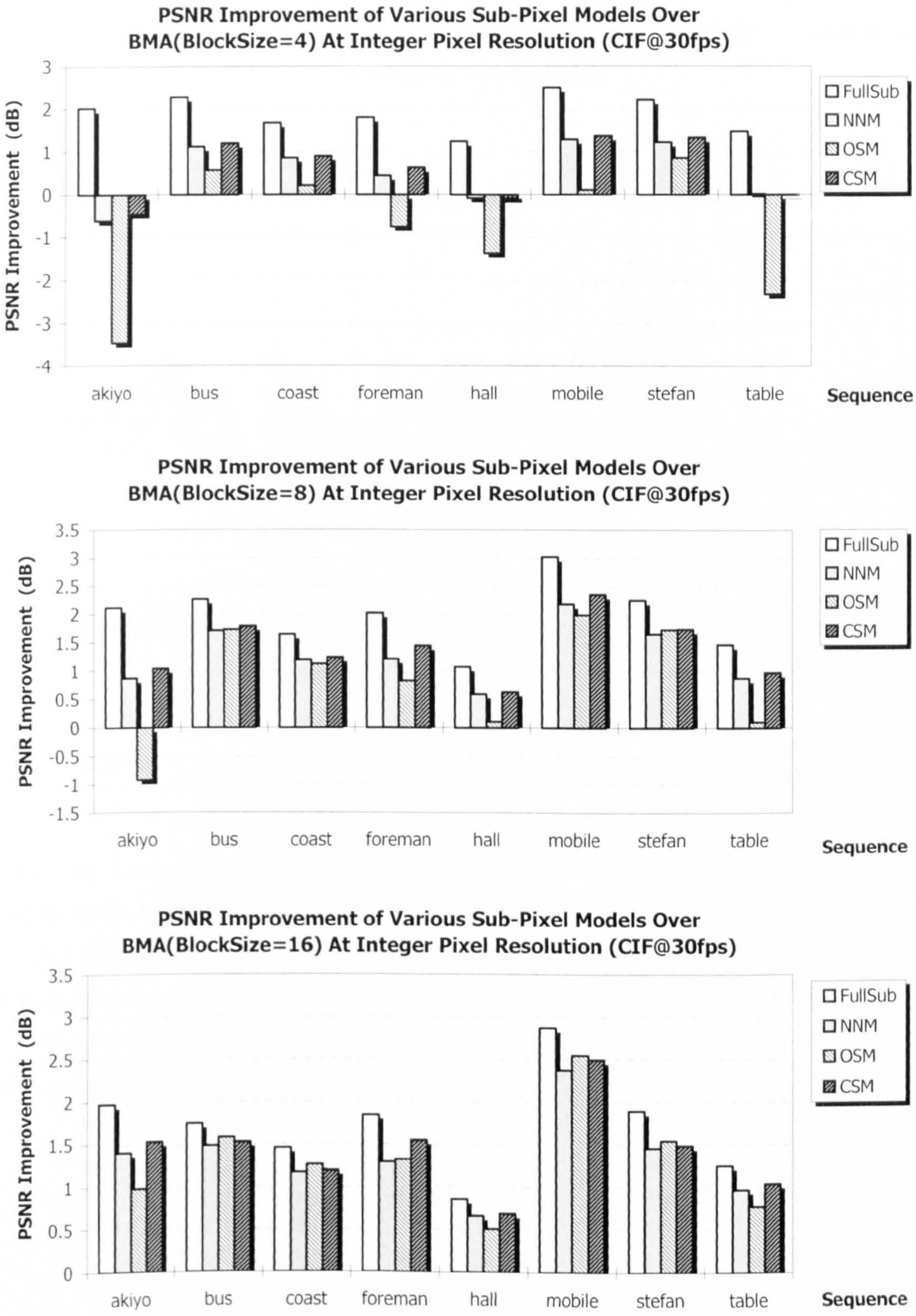


Figure 4.7 Bar charts showing the improvements of various sub-pixel model over integer-based BMA. Sequences are CIF at 30 fps using: (top) 4x4 blocks (middle) 8x8 blocks and (bottom) 16x16 blocks.

To illustrate how PSNR varies across the frames, a sample sequence of FOREMAN.CIF@30fps is shown in Figure 4.8. Despite the cluttered appearance of the graph, one observation stands out – the PSNR lines of all three model lies closer to the FullSub line than the NoSub line – indicating a more than 50% improvement over integer-pixel-based motion estimation.

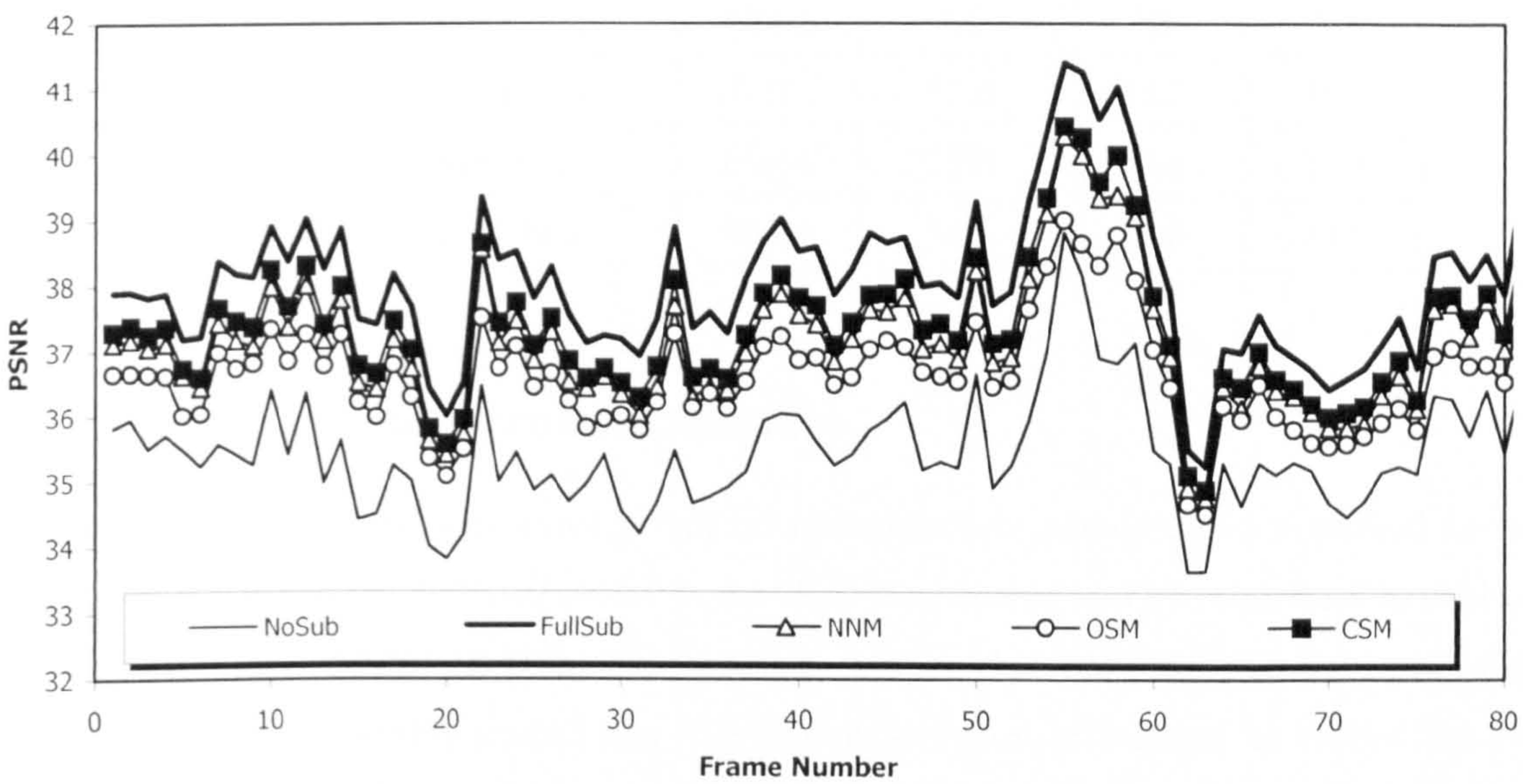


Figure 4.8 PSNR of predicted frame of FOREMAN sequence using various sub-pixel models.

In terms of the processing time required, the simulation times were averaged over all sequences and tabulated in Table 4.2. Sub-pixel refinement by all three models took approximately the same time to process. Full sub-pixel refinement via frame interpolation took an average of 18-33 times the duration of the model-based algorithms. Coupled with the large memory required to store the interpolated frames, the interpolation-free algorithms are preferred in real-time applications where processing power is limited and memory resources are scarce.

Table 4.2 Average processing time per frame for various sub-pixel (1/4-pixel) motion vector estimation algorithms in milliseconds.

	FullSub	**NNM**	**CSM**	**OSM**
QCIF@10fps, 4x4 block	256.2	13.6	14.1	13.3
QCIF@10fps, 8x8 block	151.3	5.6	5.7	5.5
QCIF@10fps, 16x16 block	124.2	3.6	3.6	3.5
CIF@30fps, 4x4 block	1027.7	55.0	58.2	55.7
CIF@30fps, 8x8 block	608.4	23.0	23.8	23.1
CIF@30fps, 16x16 block	497.4	14.5	14.6	14.5

4.1.4 Conclusions and Recommendations

From the simulation results, it is evident that interpolation-free methods give a marked savings in processing time. Due to its ability to model asymmetric distribution and its robustness towards noise, CSM is preferred over NNM and OSM. From the observation that 4x4 blocks has less gain over 8x8 and 16x16 blocks, it is recommended that CSM-based sub-pixel refinement be carried out on 8x8 blocks, and when 4x4 blocks are required (as in the case of H.264 compression), a small region around the current 8x8 displaced block in the reference frame be interpolated and used for sub-pixel refinement. Further reduction in complexity can be achieved if, instead of carrying out model-based sub-pixel refinement at integer-pixel resolution, the initial full search algorithm stops at a coarser resolution. For instance, we can conduct a full search to double-pixel resolution and then use CSM to provide further refinement. In addition, with the help of OSM, we can estimate how reliable the model parameters are from the mean square error of the solution to Eq 4-7. Then adaptive sub-pixel refinement can be performed using either frame interpolation or model-based estimation.

The proposed sub-pixel estimation method works in the SAD space. The main motivation behind working with this space is that most existing software- and hardware-based video coding systems use it. It is straight-forward to migrate the whole algorithm to other error space like mean-square-error or even block correlation. In the latter case, matching can even be done in the frequency domain with phase correlation [For-02]

4.2 Reliability Measures for the Block-Matching Algorithm

As motion estimation is plagued by problems of aperture and occlusion as well as other sources of noise, each motion vector evaluated may or may not be induced by the actual motion of the region. Figure 4.9 illustrates the variation of reliabilities within a frame. The frame consists of a global panning

motion, which is picked up consistently by BMA around the spectator background. The grass court region, however has very poorly estimated motion vector due to its lack of any texture for reliable motion estimation. Various papers use different means to represent how reliable a block motion vector is. Although the direct use of the reliability of a block may not be apparent in video compression systems, it will be used later in the setting of priority-queue-based motion estimation.

Figure 4.9 An illustration of motion estimation accuracies of different region. The frame consists of a global panning motion, which is picked up consistently by BMA around the spectator background. The grass court region, however has very poorly estimated motion vector due to its lack of any texture for reliable motion estimation.

4.2.1 Common Reliability Measures

In [Wan-00], Wang et al classifies confidence measures into (i) spatial; (ii) temporal and (iii) texture confidence measures. The first two make use of the properties of the motion vector field. Spatial reliability refers to the smoothness of the motion vector field; it measures how similar the motion vector of the current block is compared to its neighbours'. This is in line with the assumption that motion vector fields are piece-wise smooth, but this measure is not accurate at object boundaries. Temporal reliability is based on the postulate that motion vectors do not change rapidly with time. However, in sequences with low frame rate or containing fast moving objects, the measure is entirely useless.

Conversely, texture reliability, is more useful and different versions have appeared in the literature. Texture reliability is based on the assumption that uniform regions produce bad solutions to the local optical flow equations. [Hil-01] uses intensity derivatives as a fast and simple means to estimate the

reliability of the BMA vector. [Wan-00] uses a subset of the AC coefficients from the DCT transform to denote texture. In their paper, uniform regions like sky and pavements can be marked out as unreliable motion field regions. [Yos-97] uses a more complicated version of texture capable of identifying "flat", "non-flat" and "edged" blocks and assign them appropriate reliability values.

Another class of reliability makes use of the results of BMA. Recall that BMA is the process of minimizing a block mismatch measure, say the sum-of-absolute-difference (SAD). The motion vector is then the displacement such that the SAD is minimized. This minimum value, SAD_{min}, would be a good indicator of how well the displaced block matches with the current block. Many papers used similar measures (e.g. [He-01]) as weights in various algorithms using robust statistics. [Hil-01] weighted the SAD_{min} with the average SAD_{min}'s of its neighbours. A recent paper by Patras and Hendriks [Pat-02] provides a deeper insight into the SAD distribution. They use the common assumption that the displaced frame difference is Laplacian distributed and represents reliability as how close the SAD distribution matches that of a Laplacian distribution.

Three reliability measures were investigated in their ability to representation the confidence level of the associated motion vector of block k:

$$R_1(k) = \sum_{\mathbf{p} \in B_k} \sqrt{I_x^2(\mathbf{p}) + I_y^2(\mathbf{p})} \qquad \cdots(a)$$

$$R_2(k) = \frac{1}{1 + SAD_{\min}(k)} \qquad \cdots(b)$$

$$R_3(k) = \frac{1}{1 + \sum_{j \in N_k} \left\| \mathbf{v}_k - \mathbf{v}_j \right\|} \qquad \cdots(c)$$

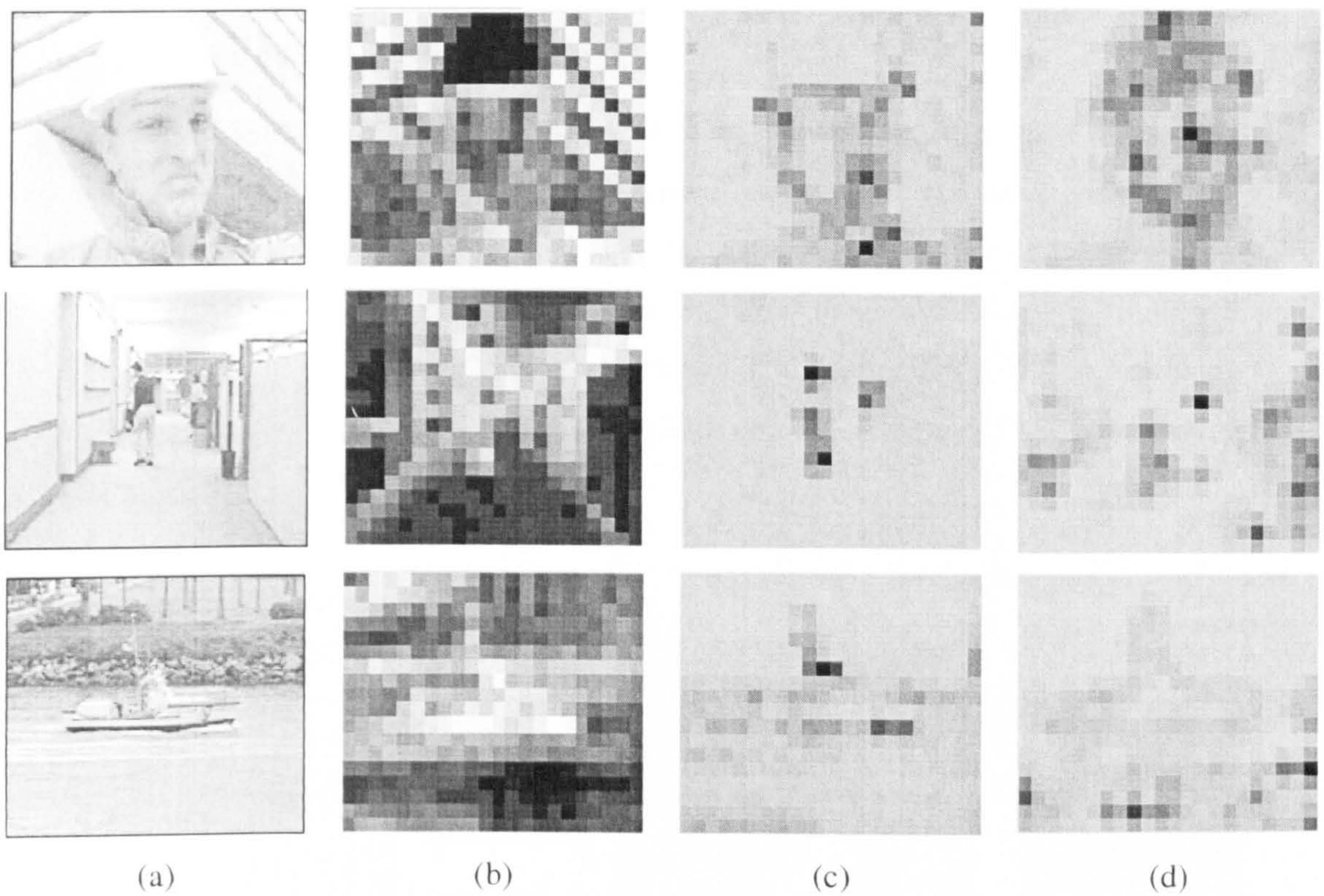
Eq 4-9

The measures in Eq 4-9 are examples of the three classes of reliability measures:

1. R_1 – texture reliability, where B_k is the set of points in block k.
2. R_2 – reliability based on minimum SAD using BMA.
3. R_3 – spatial reliability based on motion smoothness, where N_k is the indices of the neighbouring blocks of block k and $\mathbf{v}_k$ is the motion vector of block k.

To illustrate the effectiveness of the three reliability measures, Figure 4.10 shows a frame from three QCIF test sequences. The second, third and fourth columns represents R_1, R_2 and R_3 respectively. The texture reliability reflects the activity of each block, but has no direct link to the quality of the motion vector. The hard-hat region in FOREMAN.QCIF and the floor area in HALL.QCIF reflect correctly the incapability of BMA to estimate motion correctly in uniform regions. However, the edges in the top background of FOREMAN.QCIF provide good texture but motion estimated in this region is unreliable due to aperture problems. The texture reliability fails to pick up such problems. The SAD_{min} measure is very effective in identifying motion failures due to occlusion problems, as evident in the moving mask

of the ship in COAST.QCIF, revealing background as it travels along the horizontal direction. Uniform regions, on the other hand, cannot be picked up as unreliable regions when a perfect match can be found (as in the hard-hat in FOREMAN.QCIF). The motion smoothness measure is not very effective in identifying both occlusion and aperture problems.



(a) (b) (c) (d)

Figure 4.10 Reliability measures. Column (a) denotes the current input picture; column (b) represents texture reliability (R_1); (c) is the results of the SAD_{min} (R_2); (d) represents the spatial motion smoothness (R_3). The higher the intensity of the block, the larger the reliability measure.

In the next section, a novel reliability measure is introduced, which makes use of the distribution of the SAD value within the motion vector space. In contrast with the texture measure, the new reliability measure is directly linked with the BMA; the new measure is different from the SAD_{min} measure as it looks at the distribution of the SAD values instead of a single point. The novel measure has the advantage of both previous measures without their respective short falls.

4.2.2 Novel Reliability Measure, Motion Candidacy Spread

Of all reliability measures mentioned above, texture reliability and SAD_{min} are the most widely used. Texture reliability assumes a direct relation between intensity variance with the accuracy of motion estimation. Although algorithms applying such relationship have reportedly produced good estimation results, the relationship is artificial. The SAD_{min} measure uses the result of BMA and hence is a better

reliability measure. However, two blocks with equal SAD_{min} may not necessarily provide equal confidence level in terms of motion estimation.

Before the new reliability measure is introduced, the principle of block matching is revisited. This time we focus not on the minimization of SAD value, but expressing the SAD as a function of the motion vector $\mathbf{v} = (u, v) \in SW$, where $SW = [-L, L] \times [-L, L] : L \in Z^+$. SW is referred to as the search window of the BMA operation while L is the search range.

An illustration of BMA is shown in Figure 4.11. The variable k is used to represent some lexicographical index of the set of blocks in the current image and B_k the set of points in block k. The index usually follows the order of the raster scan from the top-left corner of the frame, and will be referred to as the block address. Traditional BMA is the direct process of finding the minimum point in the SAD distribution in the right-most graph of Figure 4.11 (called the SAD-map of block k). This minimum value point $\mathbf{v}_k = (u_k, v_k)$ is the motion vector and the minimum SAD value will be referred to as S^*_k of block k

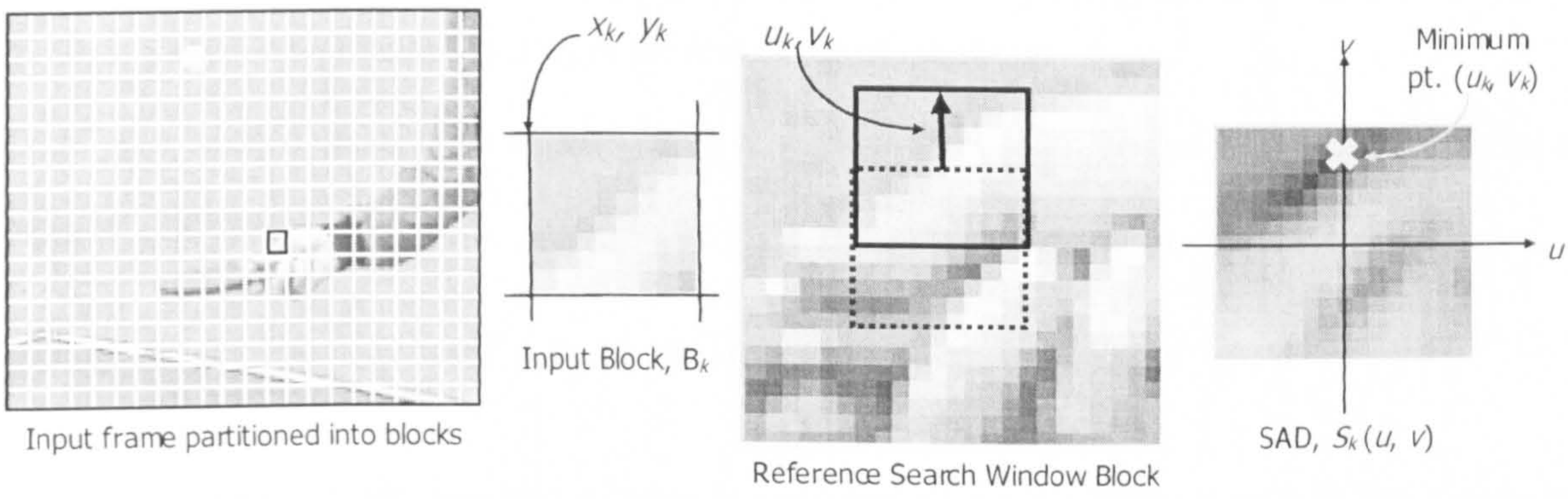


Figure 4.11 Illustration of BMA via minimization of SAD distribution.

The proposed reliability measure is based on finding a set of vectors which are potential motion vectors for each block, instead of a single minimum point. The set has to meet a few conditions:

1. The set must not be empty. That is, at least the global minimum within the block has to be a member of the candidate set.
2. The set must contain the global minimum point.
3. The cardinal of the set must be able to vary according to the SAD-map.

A few intuitive selection schemes are considered:

- $C1 = \{\mathbf{v} : S(\mathbf{v}) \le \mu_S - R_1 \sigma_S, R_1 > 0\}$, where μ_S and σ_S are the distribution mean and standard deviation of the SAD-map, respectively.
- $C2 = \{\mathbf{v} : S(\mathbf{v}) \le R_2^{\text{th}} \text{ percentile of } \mathbf{H}(S)\}$, where $\mathbf{H}(S)$ is the histogram of the SAD-map.

- $C3 = \{\mathbf{v} : S(\mathbf{v}) \leq Min(S) + R[Max(S) - Min(S)]\ \ 0 < R < 1\}$

Figure 4.12 uses three one-dimensional SAD-maps to compare and contrast the 3 selection criteria. The first column is an SAD-map with 2 minimum points; the centre column is a relatively 'flat' SAD-map while the third SAD-map has a distinct minimum point. Each candidate point is marked with a circle. All three criteria can detect the two minima in column 1. The short-coming of C1 is revealed in SAD-map 2, where there is no candidate point at all. Hence C1 is eliminated due to its failure to meet condition 1. Both C2 and C3 are guaranteed to produce at least one candidate, but as C2 is related to rank statistics, the number of candidate points is fixed. As a result C2 produces too many candidate points in SAD-map 3 which only has a distinct minimum. C3 is by far the best criterion. From Figure 4.12, the threshold value in C3 adapts itself according to the SAD distribution and produces more accurate candidacy sets in all three cases. An added advantage was discovered from the outcome of the simulation results that the candidacy set is not sensitive to the value of R when R is sufficiently small (R<0.5) for most SAD-maps. More uniform SAD-maps are more sensitive to the R value, but in such cases the candidacy set is usually large, indicating that they are unreliable blocks. As we shall see later, the ordering of unreliable blocks amongst each other are less important. In conclusion, C3 is the best candidacy criterion and it will be used in the remainder of the thesis.

Hence, with the SAD-map $S_k(\mathbf{v})$ of each block k, we identify a set of candidate vectors (candidate set) $Cand(k;R)$ defined as:

$$Cand(k;R) = \{\mathbf{v} \in [-L, L]^2 : \quad S_k(\mathbf{v}) < Min(S_k) + \quad R[Max(S_k) - Min(S_k)]\} \qquad \text{Eq 4-10}$$

R provides a threshold below which a point is considered a good motion candidate; the threshold is termed as the candidacy threshold. The value of R varies from 0 to 1, defining the amount of candidate vectors to consider. The terms $Min(\bullet)$ and $Max(\bullet)$ are the minimum and maximum values amongst S_k. We next define the amount of spread amongst these candidate points, called the motion candidacy spread (MCS):

$$Spread(k;R) = \sum_{\mathbf{v} \in Cand(k;R)} \left[\sum_{\mathbf{u} \in Cand(k;R)/\{\mathbf{v}\}} \|\mathbf{v} - \mathbf{u}\| \right] \qquad \text{Eq 4-11}$$

The spread is essentially the sum of Euclidean distances amongst all candidate vector pairs. The sum has two effects: (i) more candidate vectors produce a higher spread as more pairs are computed; (ii) given the same number of candidate vectors, dispersed set produces a large spread. Hence, the spread measures how loosely the candidate vectors are distributed. The block with a low MCS is a block with a reliable BMA result.

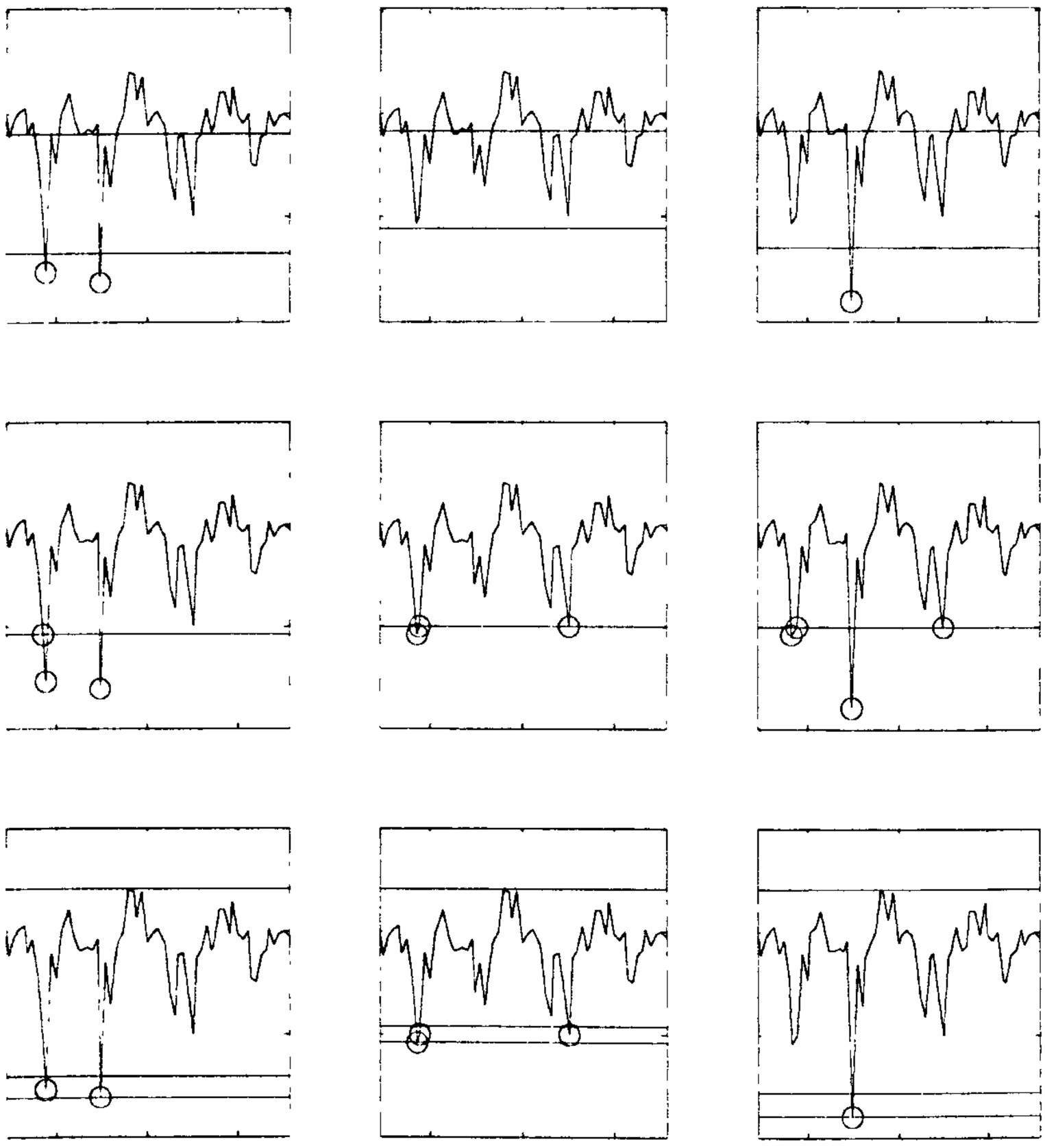


Figure 4.12 1-dimensional 'SAD-map' comparing the merits and pit-falls of 3 candidacy criteria. Column 1 – an SAD-map with 2 minimums. Column 2 – a relatively 'flat' SAD-map. Column 3- SAP-map with a distinctive minimum. Each row shows the 3 selection criteria C1, C2 and C3 respectively. Solid lines show the respective threshold levels; dotted line in C1 charts is the mean, and the 2 dotted lines in C3 charts represents the minimum and maximum values.

The reliability of each block is taken as the reciprocal of S_k. This reliability measure can detect unreliable blocks due to occlusion and aperture problems. It also identifies blocks containing multiple objects moving in different directions.

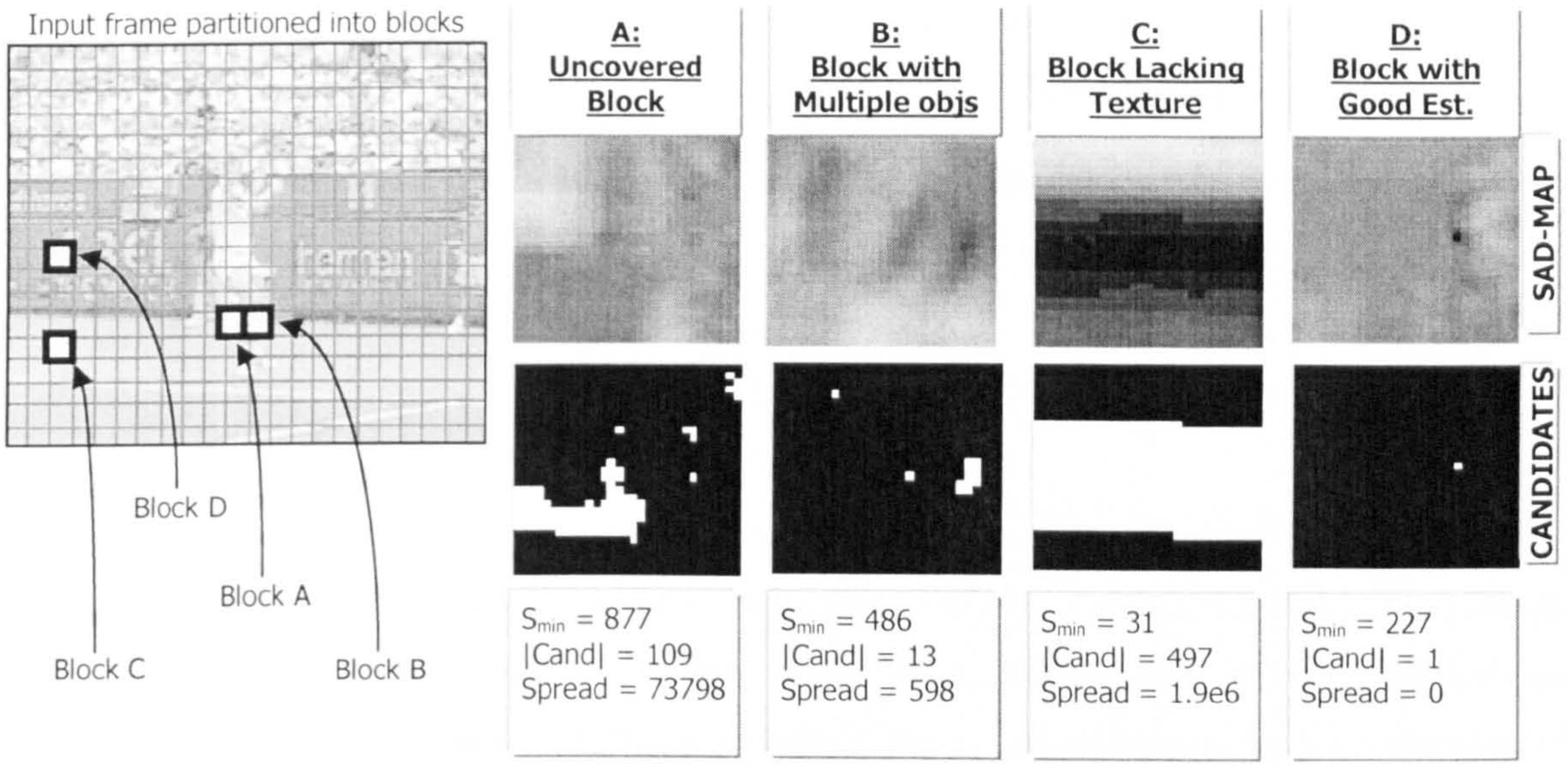


Figure 4.13 Illustration of the effectiveness of motion candidacy spread (MCS).

Figure 4.13 illustrates the effectiveness of the novel reliability measure by MCS. Block B contains multiple objects, which manifests itself as small multiple disconnected candidacy regions. The spread measure within each connected region is small; however, the global spread, taken into consideration inter-region spread, becomes large. In block C, due to the lack of texture, the minimum SAD value is very small. If the SAD_{min} measure is used, this block would have been a reliable block, contrary to the observation. But the MCS measure successfully identifies this block as very unreliable. Block D has a good motion spread in spite of a relatively high SAD_{min} (probably due to the quantization of the motion vector, or a change in lighting conditions); only one candidate point is identified, thus giving a zero MCS. Uncovered regions like block A gives very high SAD_{min}, which can also be identified with our new measure.

In the next section, a queue-based algorithm is introduced to incorporate smoothness constraint into the BMA, which produces a more natural motion vector field than the traditional BMA. The smoothed motion field produces a better starting point for global motion estimation and motion segmentation; it is also a good alternative to BMA with normal raster scan, where motion vectors are predictive-coded causally from previously encoded blocks.

4.3 Implementation of Smoothness Constraints

In traditional BMA, motion vectors evaluated are the result of the minimization of SAD maps. For blocks with little texture, the SAD distributions are relatively flat and more than one vector may produce similarly low residues. By selecting the motion vector strictly based on the minimum SAD

values, the residual energy may be minimized at the expense of increasing entropy of the motion vector field. By introducing smoothness constraints to the motion vector field, the coding overhead of the motion field can be reduced. In some cases, this is similar to solving the aperture problem.

Traditional BMA builds up a sparse motion vector field evaluating the $\{\mathbf{v}_k\}$ set using Eq 4-12.

$$S_k(\mathbf{v}) = \sum_{\mathbf{p} \in B_k} \left| I_t(\mathbf{p}) - I_{t-1}(\mathbf{p}+\mathbf{v}) \right| \qquad \text{Eq 4-12}$$

$$\mathbf{v}_k = \arg\min_{\mathbf{u} \in [-L,+L]^2} S_k(\mathbf{u})$$

Unfortunately, some blocks are very prone to noise due to clutter, occlusion and the lack of texture. This results in a somewhat erroneous motion vector map, which makes it unsuitable for image registration applications and incurs extra bits in coding these vectors. Smoothing via post processing e.g. median filtering improves smoothness globally, but this tends to over-smooth object boundaries.

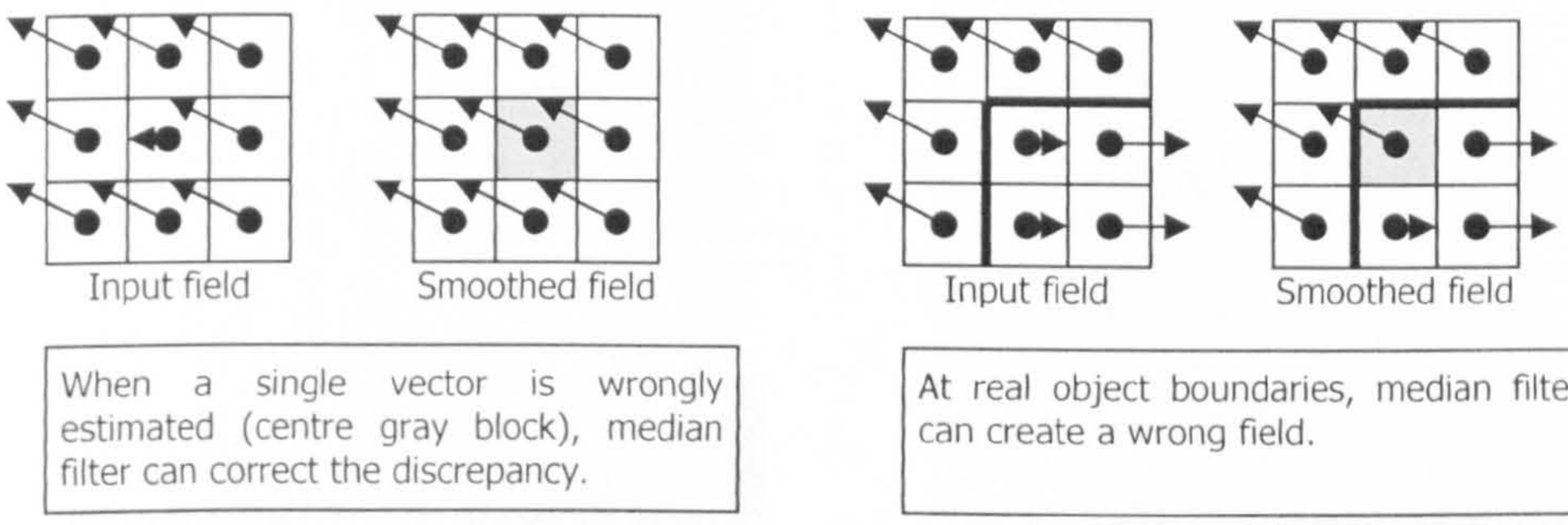


Figure 4.14 Illustration how median filter can create wrongly smoothed fields.

Other methods of implementing global smoothness using statistical relaxation methods (e.g. simulated annealing and iterative conditional modes) produces more optimal results, but these methods are highly computationally intensive and are not suitable for real-time implementation.

We propose to implement the smoothness constraint with reasonable computation complexity via a modified cost measure $T_k(\mathbf{v};\mathbf{v}_p)$ which adds to $S_k(\mathbf{v})$ an extra penalty proportional to the amount $\mathbf{v}$ deviates from a predictor $\mathbf{v}_p$.

$$T_k(\mathbf{v};\mathbf{v}_p) = S_k(\mathbf{v}) + \lambda \left| \mathbf{v} - \mathbf{v}_p \right| \qquad \text{Eq 4-13}$$

The parameter λ is the relative weight of the smoothness constraint with respect to the original cost S_k. To maintain spatial smoothness of the vector field, the predictor $\mathbf{v}_p$ is obtained from the neighbouring

block. This thesis uses the 4-way connectivity neighbourhood system, η_k as shown in Figure 3.2. Hence in place of Eq 4-12, we use Eq 4-14 to introduce motion smoothness to BMA:

$$\mathbf{v}_k = \arg \min_{\substack{\mathbf{v} \in SW \\ \mathbf{v}_p \in \eta_k}} T_j(\mathbf{v}; \mathbf{v}_p) \qquad \text{Eq 4-14}$$

To illustrate the effect of Eq 4-14 on BMA, Figure 4.15 shows the original SAD map of a block without a distinct minimum point (left) and a modified SAD-map which is offset by the distance from (-2, -5). The constraint introduces a distinct minimum point, which minimizes the motion residues as well as textural residues. In the blocks whose SAD_{map}'s have distinct minimums, the constraint would not 'displace' the original minimum points if a proper λ value is chosen.

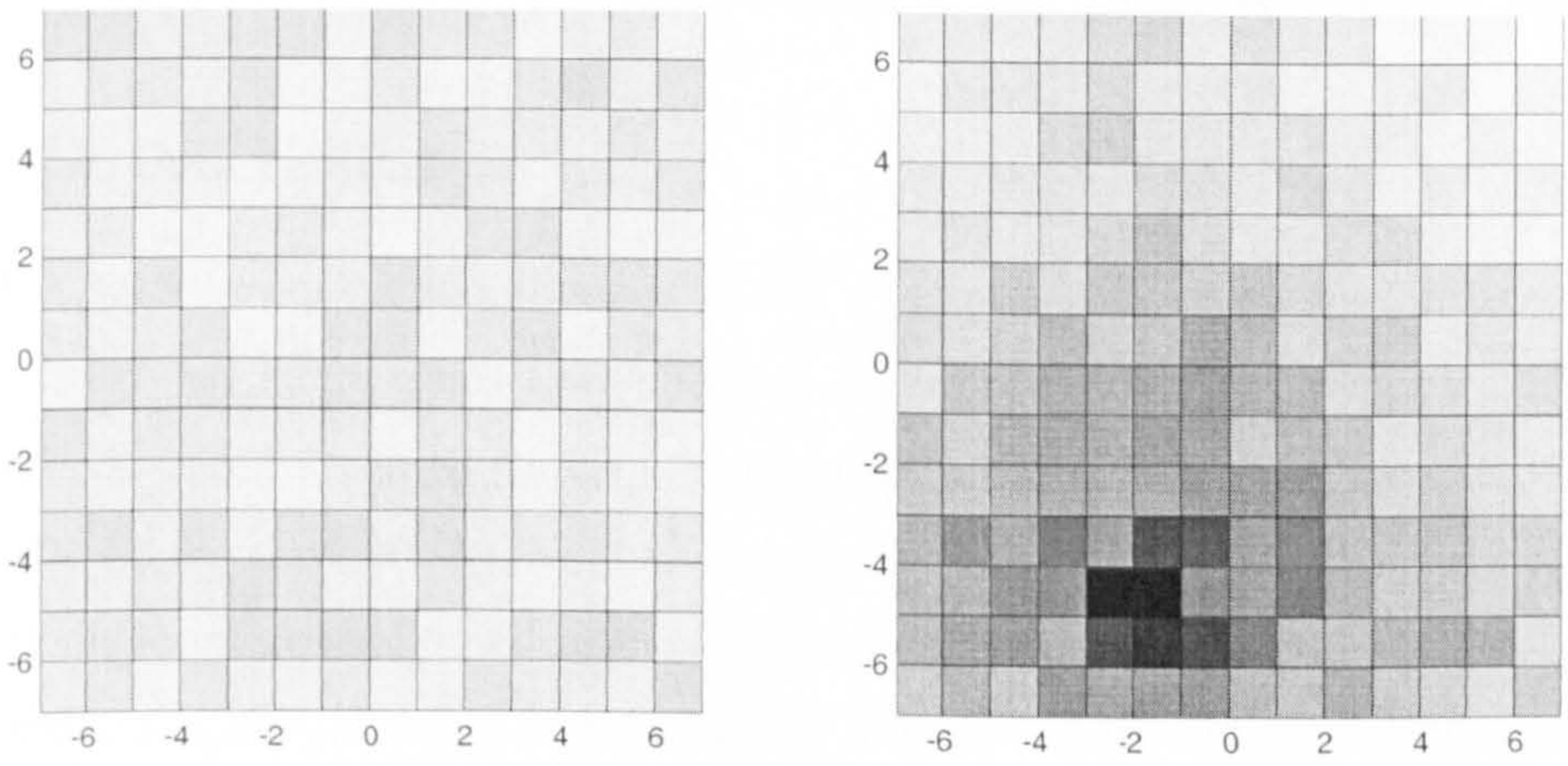


Figure 4.15 Illustration of smoothness constraint. By constraining the flat SAD-map towards a neighbouring block with motion vector at (-2, -5), the modified SAD-map shows a distinct minimum SAD region, the minimum point produces a motion vector closer to its neighbour's.

It is worth mentioning that all the common video compression standards (e.g. H.263 and H.264) codes motion vectors differentially based on a predictor. The predictor used in each block is usually some central measure (usually medium) of a set of motion vectors of neighbouring blocks which are already processed. In normal raster scanned blocks, these are the blocks to the left and above the current block. The detailed algorithms in selecting the predictor set is more complicated in the standards due to special cases like blocks at the borders and blocks with Intra-coded neighbours which lacks a motion vector. Please refer to the respective standards for details. Most encoders would process the block at the top-left corner first, then process the other blocks in raster-scan order. Motion estimation of subsequent blocks is done in favour of the predictor derived from its processed neighbours. This constitutes a form of smoothness constraint. It is not difficult to see that such constraint is not optimal. The first blocks

being processed are the top- and left-most boundary blocks. Motion vector found from these blocks are known to be highly unreliable, as the 'true motion' may lie beyond the picture and hence cannot be evaluated. Motion vectors, when subsequently used as predictor, may sway motion vector of the following blocks further away from the actual motion vectors. Hence, smoothness constraints should be implemented with a slightly more 'global' perspective, without the restriction of the raster scan. The queue-based motion estimation discussed below does just that. A priority queue is set up in the order of how reliable the motion vector is. Motion of the most reliable blocks which do not need to be constrained by the neighbouring blocks is processed first. Subsequent blocks are then 'constrained' according to their 'processed', more reliable neighbours. The resulting motion vector field is found to be smoother and more natural visibly. It also carries less entropy without introducing excessive extra residual energies.

4.4 Queue-Based Motion Estimation with Smoothness Constraints

4.4.1 QBMA Description

As explained in the previous section, implementation of a smoothness constraint in a causal manner is not ideal. On the other hand, global minimization of the problem via simulated and deterministic annealing methods can be computationally intractable. Even sub-optimal solutions like gradient-descent may still be too complex. In this thesis, the queue-based BMA (QBMA) method is proposed. This is a novel single-pass queue-based method which utilizes the concept of reliability and smoothness constraint described in the preceding sections. Being a deterministic and non-iterative algorithm, it is robust and fast convergence is guaranteed.

Instead of performing block motion estimation in the usual raster-scan order, a priority queue is used to sequence the BMA process. Based on the motion candidacy spread (MCS) described above, the priority queue is set up in descending order of the spread measure. Using MCS to dictate the order of BMA, blocks with a reliable SAD-map are processed without any motion constraint. Neighbouring blocks which are processed subsequently will have their motion vectors constrained to their neighbouring blocks that have been processed. Hence unreliable blocks can peg their motion vector towards its more reliable neighbours. This smoothness constraint, however, preserves true motion edges as the SAD term dominates the smoothness constraint when there is a true object boundary.

The algorithm is described as a flow chart in Figure 4.16:

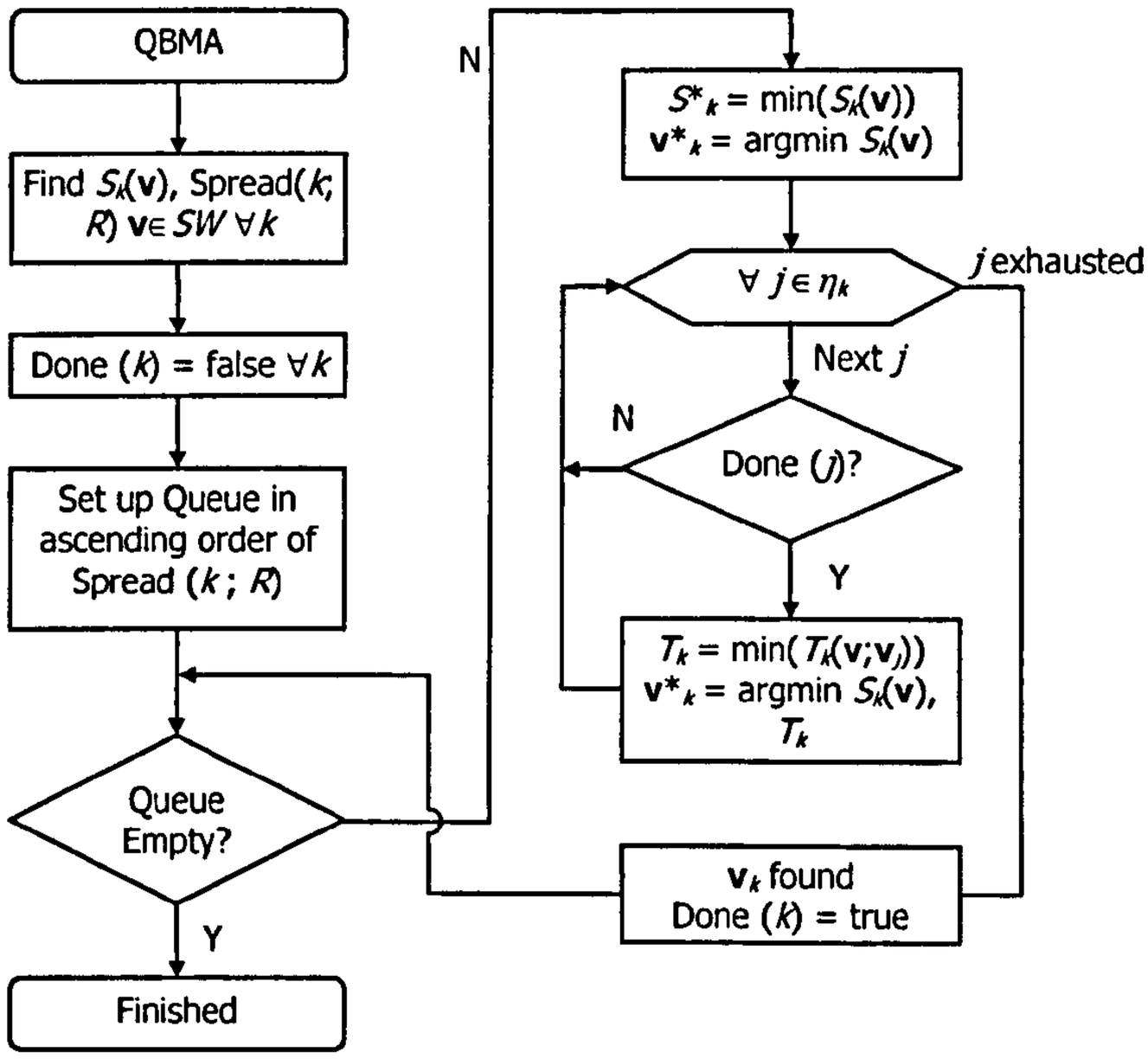


Figure 4.16 The flow chart of queue-based BMA (QBMA).

The first step in QBMA is the evaluation of the SAD map for each block of image. The SAD map is a distribution of SAD values with respect to motion vectors. Without reference frame interpolation, the SAD map can reach one-pixel resolution. If processing power is limited, SAD maps of lower resolution can be used. With typical test sequences, the SAD map is usually globally smooth such that a sub-sampled SAD map is sufficient for obtaining an accurate ranking of the blocks' reliabilities.

As an aside, evaluation of the SAD map followed by finding the minimum location is procedurally equivalent to a full-search based BMA. In strictly serial computational systems, there is an optimization step in the full-search BMA which is not possible in SAD-map evaluation. As full-search aims at finding the candidate motion vector giving the minimum SAD, evaluation of SAD values of other vectors can stop prematurely. The moment the sum exceeds that of the minimum SAD, evaluation of current SAD can stop and the full-search can proceed to the next vector. With this optimization, the computational load of the full search algorithm would be hypothetically much lower than that required by the process of evaluating the SAD map. In reality, the reduction does not exist in most real-time applications. Firstly, most hardware-based systems are parallel in nature and SAD values are evaluated simultaneously; in such cases, premature exit from the summation loop is not possible. Secondly, even in processor-based system, the highly pipelined architectures which brings about many-fold increase in processing speed is very inefficient in handling conditional loops. These loops, which are used in full-search algorithms to test for exit conditions, break the pipeline and should be avoided. In the smaller

block sizes like 4x4, it would be faster to evaluate the full SAD than to check for such conditions. These two given reasons forms the basic justification of the using the SAD-map for BMA.

The SAD-map for each block is used to produce the motion candidacy spread (MCS) measure (see 4.2.2). The MCS serves as a reliability measure for the block's motion vector and determines the relative order in which the blocks are processed subsequently. To facilitate the processing of blocks in the order of their reliabilities, a priority queue is formed; block addresses are pushed in descending order of the blocks' MCS. At the same time, a two-dimensional Boolean Done-map is set up, which has the same dimension as the blocks partition of the image. A 'TRUE' value in a location of the Done-map signifies that the current block has been processed. All Done-map locations are initialized to FALSE, and once a block is processed, the corresponding Done-map location is set to TRUE. When a block is being processed, the Done-map is inspected to identify 'processed' neighbours. Smoothness constraint as described in 4.3 is then used using the processed neighbours.

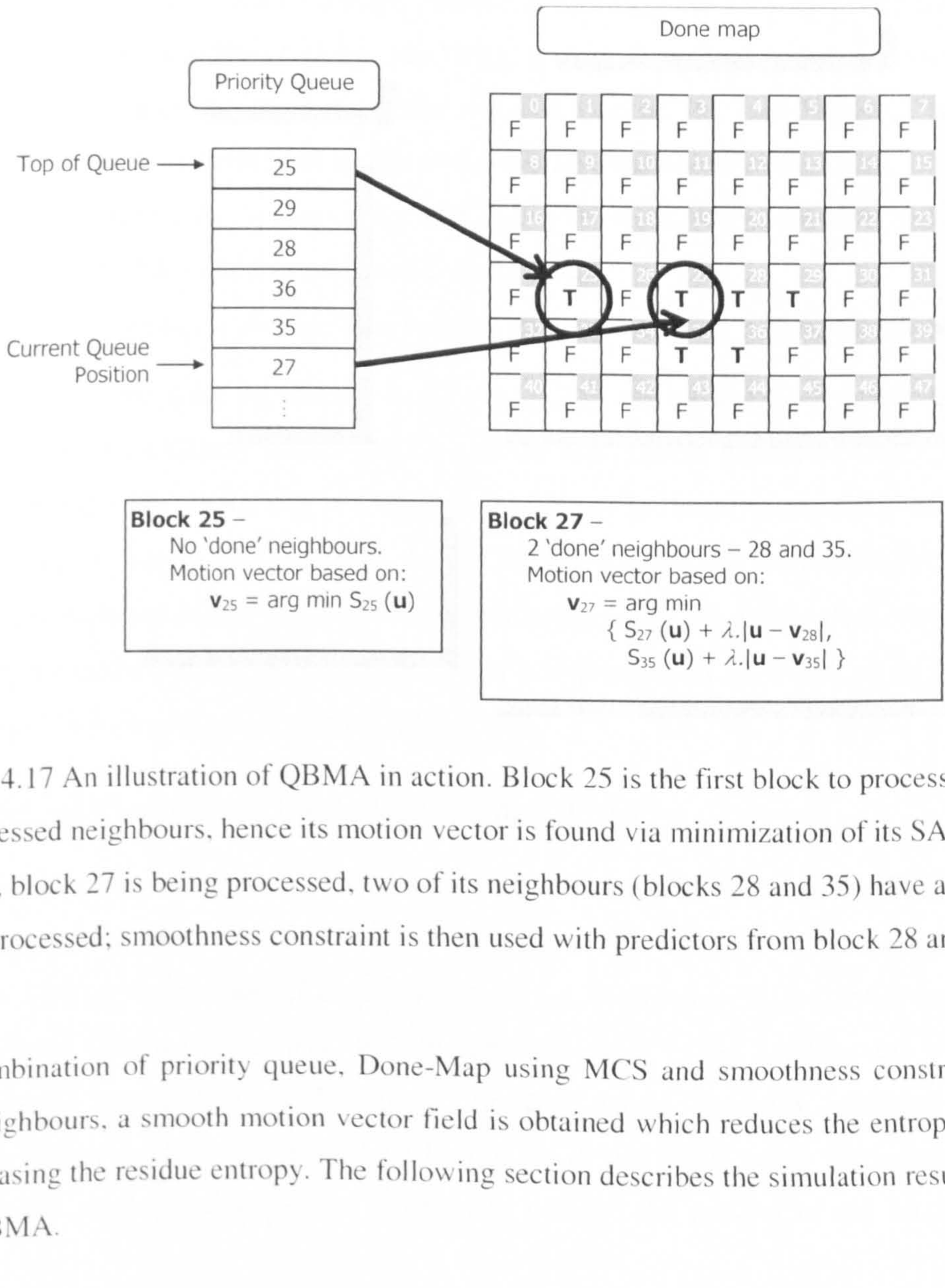


Figure 4.17 An illustration of QBMA in action. Block 25 is the first block to process. It has no processed neighbours, hence its motion vector is found via minimization of its SAD-map. When, block 27 is being processed, two of its neighbours (blocks 28 and 35) have already been processed; smoothness constraint is then used with predictors from block 28 and 35).

With the combination of priority queue, Done-Map using MCS and smoothness constraint with the processed neighbours, a smooth motion vector field is obtained which reduces the entropy of the field without increasing the residue entropy. The following section describes the simulation results justifying the use of QBMA.

4.4.2 QBMA Simulations Results and Conclusions

Results from QBMA and the traditional full-search BMA (TBMA) on eight QCIF and CIF test sequences of 10-second duration are obtained from simulation. As in the previous simulation, QCIF sequences are sampled at 10 fps (frames per second) and CIF sequences at 30 fps; block sizes of 4, 8 and 16 are used.

Each simulation produces a set of measures:

- The residue entropy, E_{Res} – measures the amount of energy left after the motion estimation. This is an indication of the upper limit of the number of bits required to code the DFD (displaced frame difference) before lossy compression. It also measures how good the motion vector field predicts the current frame from the previous frame; a larger value indicates lower prediction efficiency.
- The motion vector entropy, E_{Mv} – measures the number of bits required to code the motion vectors. To have a measure consistent with E_{Res}, the motion vector entropy is divided by the number of pixels per block, giving an entropy value per pixel. This measure indicates the number of bits required to code the motion vector in pixel-resolution. It also provides an insight to the overall smoothness of the field.
- The total entropy, E_{Tot} – sum of E_{Mv} and E_{Res}. The total entropy combines the number of bits required to code the residue and vector field.
- Processing time t_{Proc} – average time in milliseconds required to process per frame.

The QBMA has two operating parameters (see Figure 4.16): the candidacy threshold ratio R used to determine the motion candidacy spread (MCS) and the smoothness constraint factor λ in Eq 4-13. Candidacy threshold ratio R values of 0.1, 0.3 and 0.5 are used. These values will produce increasing number of candidate points, and possibly the MCS. The smoothness constraint factor values (λ) determines the importance of the smoothness constraint relative to the SAD values. As the value of λ increases, E_{Res} increases and E_{Mv} decreases. Hence, it is expected that there will be an optimum value for each sequence which will give a minimum E_{Tot} value.

4.4.2.1 Effect of Candidacy Threshold Ratio on QBMA

Across the three block sizes and five λ values, simulations results from all of the CIF and QCIF sequences indicate that there is little effect of R on the total entropies. This can be attributed to two main reasons. Firstly, the number of candidate points for reliable blocks are very insensitive to the value of R. These blocks have a very distinct minimum point in the SAD_{map}, and it requires that $R \approx 1.0$ to increase the cardinal of the set of candidacy points. Secondly, although the MCS of unreliable blocks are more sensitive to the variation of R, the queue-based implementation is only dependent on the relative values MCS; to be more precise, the ranking of the blocks' MCS. By

increasing the value of R, candidacy sets of unreliable blocks may change (and so do their MCS values), the relative reliability of the blocks remains unchanged as a whole. Hence the order in which the blocks are processed essentially stays the same with changing R. Of course, in some cases a few orders may have changed, these belong to the very unreliable blocks and they would have motion constrained by their more reliable neighbours. In conclusion, we have tested all sequences (with varying block sizes and smoothness constraint factors) using R values of 0.1, 0.3 and 0.5. The total entropies E_{Tot} of the sequences are shown not to vary significantly with R. As a result, we can safely focus on other observations of our simulation using $R = 0.1$. The independence of the entropies on the R value makes a strong case for MCS as a reliability indicator.

4.4.2.2 Effect of Smoothness Constraint Factor on QBMA

In contrast to the invariant property of the compression efficiency with the candidacy threshold ratio, the former does vary with smoothness constraint factor (λ). This section explores the combined effect of the smoothness constraint factor, block size, image size on motion estimation across the various test sequences. The Efficiency of QBMA is measured by the reduction of total entropy (the sum of motion vector entropy and residue entropy)

Firstly, the general trend of coding efficiency against changing smoothness constraint factor (λ in Eq 4-13) is investigated. An example is illustrated in Figure 4.18. Coding efficiency of the QBMA is represented by the reduction of total entropy with QBMA using a particular λ value with respect to that achieved by traditional full search BMA. The reduction in total entropy is represented as ΔE_T. Henceforth, the chart will be referred to as $\lambda - \Delta E_T$ curve. A few salient points need to be pointed out. Firstly, the $\lambda - \Delta E_T$ curve passes through the origin. At $\lambda = 0$, the smoothness constraint factor is not used and QBMA degenerates to the full-search BMA, thus the entropy is simply the entropy of the BMA, hence $\Delta E_T = 0$. Secondly, a positive value of ΔE_T indicates an improvement of QBMA coding efficiency over BMA; a negative value indicates that with current λ value, the total entropy achieved by QBMA is higher than that achieved by BMA. Figure 4.18 shows the typical shape of a $\lambda - \Delta E_T$ curve. It is generally convex and increases from the origin and peaks at a certain λ value. After the turning point, the curve becomes decreasing. At some point, the curve crosses the horizontal axis and becomes progressively negative. The part of the $\lambda - \Delta E_T$ curve with positive curve represents the values of λ giving increasing efficient QBMA results; within this range motion vector entropy can be reduced by more smoothing without introducing excessive residue. After the maximum point, increasing λ causes the QBMA to produce a larger increase in the residue entropy than the decrease in motion entropy. As the $\lambda - \Delta E_T$ curve crosses the x-axis, QBMA 'over-smooth' the field and becomes less efficient than BMA. Hence, QBMA operates best at the maximum point, but this point varies with numerous factors,

amongst which includes block sizes, picture sizes and picture content. A more realistic target would be to ensure that QBMA operates within the region of the $\lambda - \Delta E_T$ curve where it remains positive.

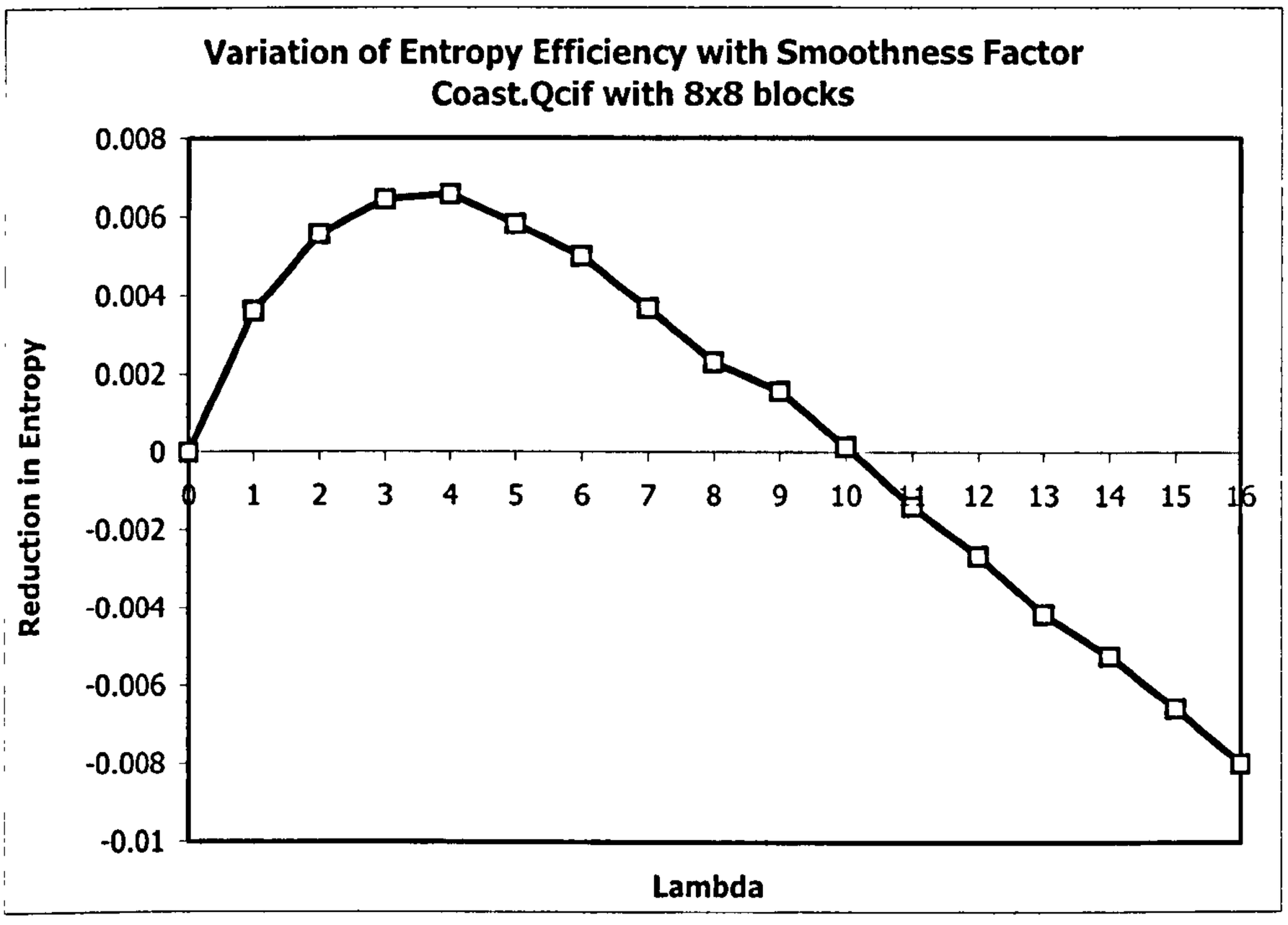


Figure 4.18 Chart showing variation of coding efficiency of QBMA with respect to smoothness constraint factorλ. Coding efficiency of the QBMA is represented by the reduction of total entropy with QBMA using a particular λ value with respect to that achieved by traditional full search BMA.

The remaining part of this section determines how this desirable operating region varies with different sequences, picture sizes and block sizes.

4.4.2.3 Effect of Picture Size on QBMA

First, the effect of picture size on the $\lambda - \Delta E_T$ curve is investigated. Figure 4.19 depicts 4 charts each containing 2 $\lambda - \Delta E_T$ curves of QBMA results on CIF and QCIF versions of the same sequence using the same block size. Note that the ranges of λ used in the simulations of 4×4 blocks, 8×8 blocks and 16×16 blocks are different. Specifically, the ranges are $[0,4]$, $[0,16]$ and $[0,64]$ respectively. This range was selected as a result of a preliminary simulation with a coarse set of values $\{0.0, 0.25, 1.0, 4.0, 16.0, 64.0\}$on all three block sizes. It is evident from Figure 4.19 that the peak values of the $\lambda - \Delta E_T$ curves vary amongst the charts. On the other hand, the CIF and QCIF curves in the same chart peak approximately within similar regions. This forms the premise that a single

favourable operating range exists in both CIF and QCIF pictures of the same sequence, provided the same block size is used.

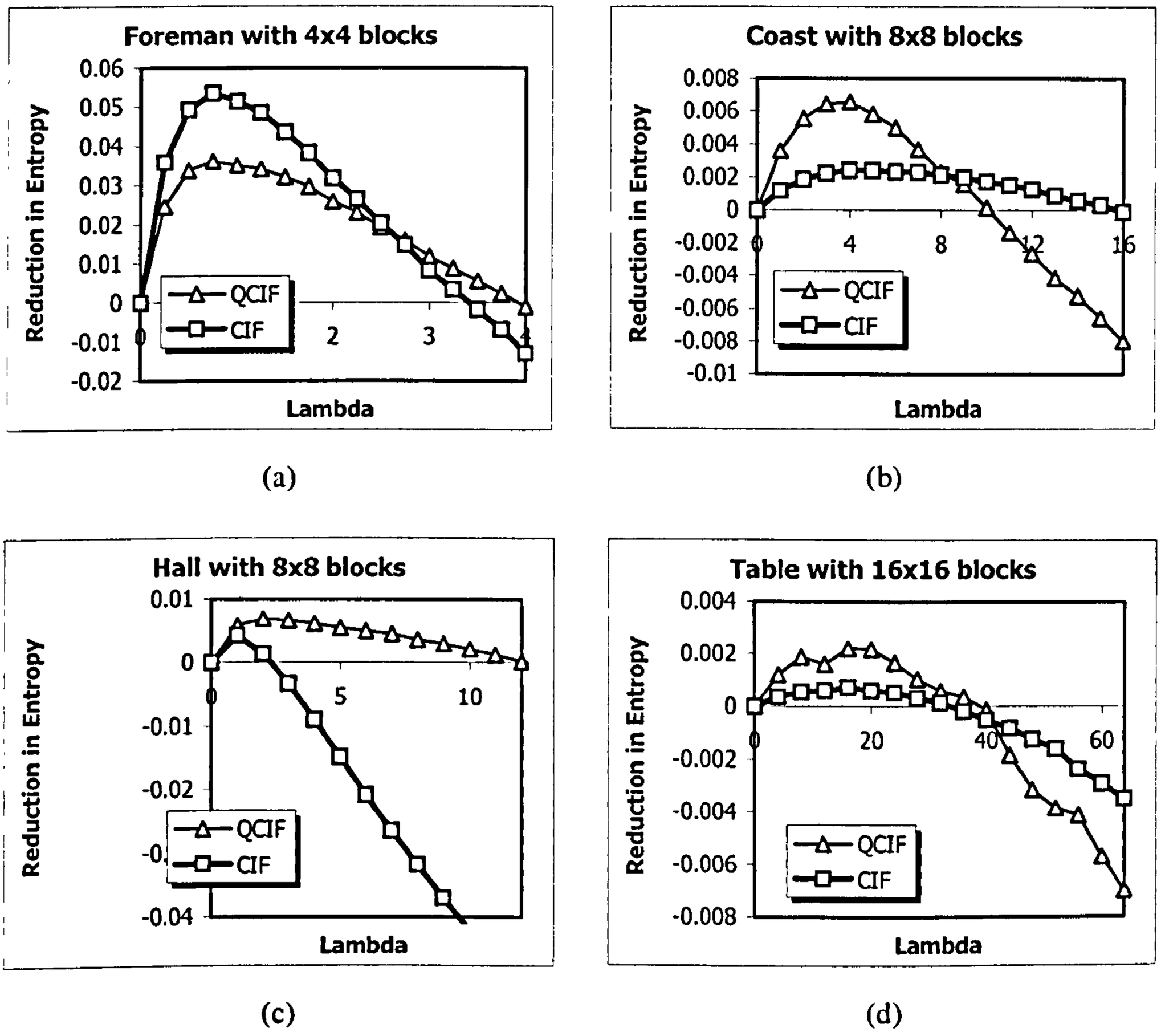


Figure 4.19 Four charts depicting the variation of QBMA efficiency with smoothness constraint factor of the same sequence at different picture sizes (CIF@30fps and QCIF@10fps).

4.4.2.4 Effect of Block Size on QBMA

Having determined that the operating range of λ in QBMA is similar in CIF and QCIF of the same sequence, we next explore how this range changes across different block sizes. Figure 4.20 shows four charts, each containing three $\lambda - \Delta E_T$ curves of applying QBMA on selected QCIF and CIF sequences with varying block sizes. As expected all curves exhibit a similar convex shape. In all four charts, the $\lambda - \Delta E_T$ curves belonging to the largest block size (16×16) are flatter than the rest; at the same time, they show much lower peaks, indicating QBMA produces less coding gain compared with 8×8 blocks and 4×4 blocks. This is due to the fact that larger blocks contains more structure for the

traditional full search BMA to produce a better match, thus is less susceptible to the aperture problem. Hence the smoothness constraint has less effect on motion vector field.

On the other hand, all four charts in Figure 4.20 show that the $\lambda - \Delta E_T$ curves belonging to the smallest block size (4×4) have higher peaks but narrower range with possible ΔE_T values. It can hence be deduced that QBMA brings more coding gain when block sizes are small; however, care must be taken in selecting λ to ensure its value falls within the favourable operating range.

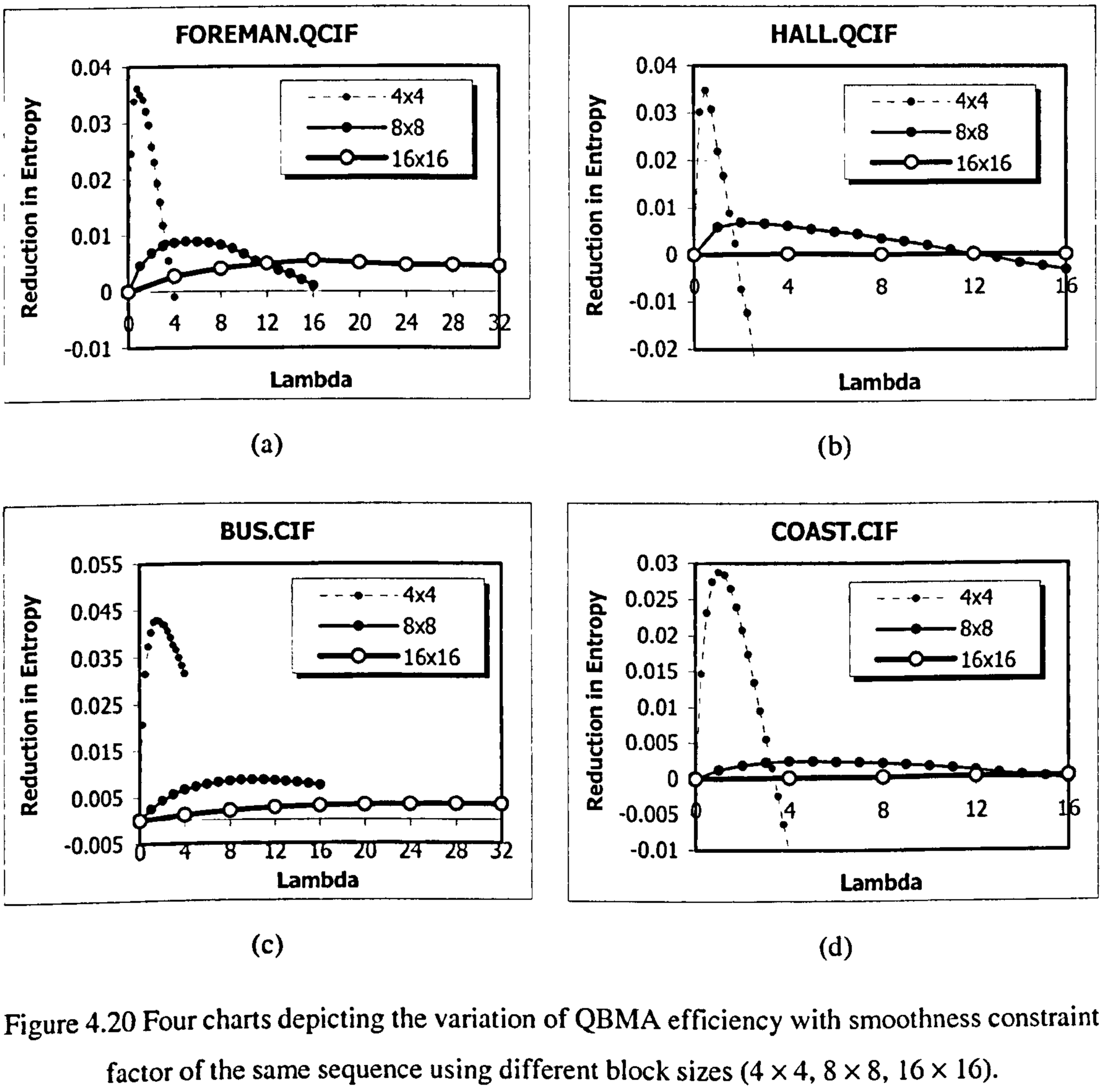


Figure 4.20 Four charts depicting the variation of QBMA efficiency with smoothness constraint factor of the same sequence using different block sizes (4×4, 8×8, 16×16).

In conclusion, the variation of $\lambda - \Delta E_T$ curves with block sizes can be summarised in Figure 4.21. As block size increase, the $\lambda - \Delta E_T$ curve stretches rightwards and is compressed vertically. As a result, the range of λ values which produce positive ΔE_T (depicted as $R_{n\times n}$ where n is 4, 8, or 16 in Figure 4.21) grows and the peak coding gain ($\hat{E}_{n\times n}$) drops and the optimal operating value of λ ($\lambda_{n\times n}$) shifts to

the right. As $\lambda_{n\times n}$ changes with block sizes, it is essential that a meaningful relationship be established. This decision will be postponed until the next section, where we explore how $\lambda - \Delta E_T$ curves changes across different pictures.

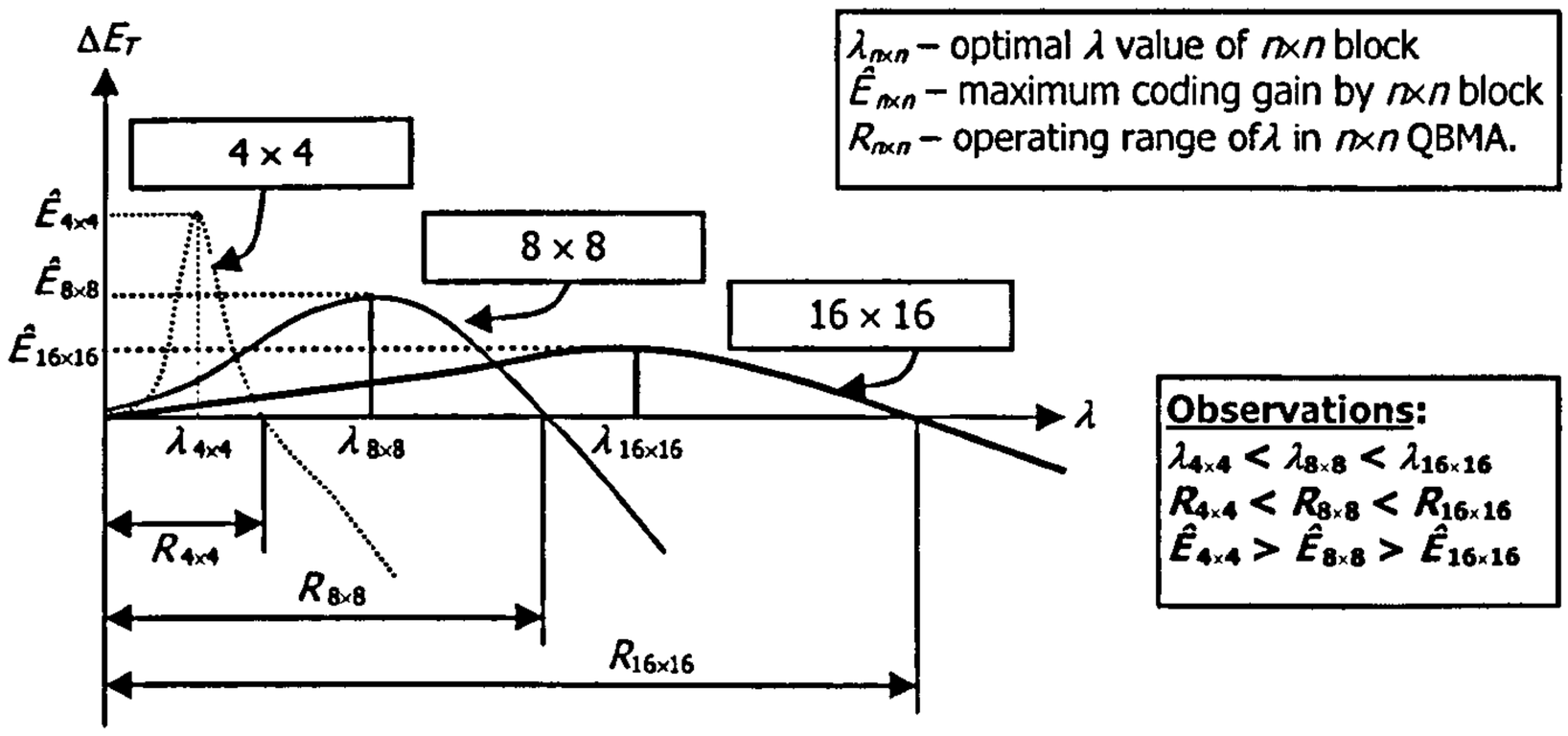


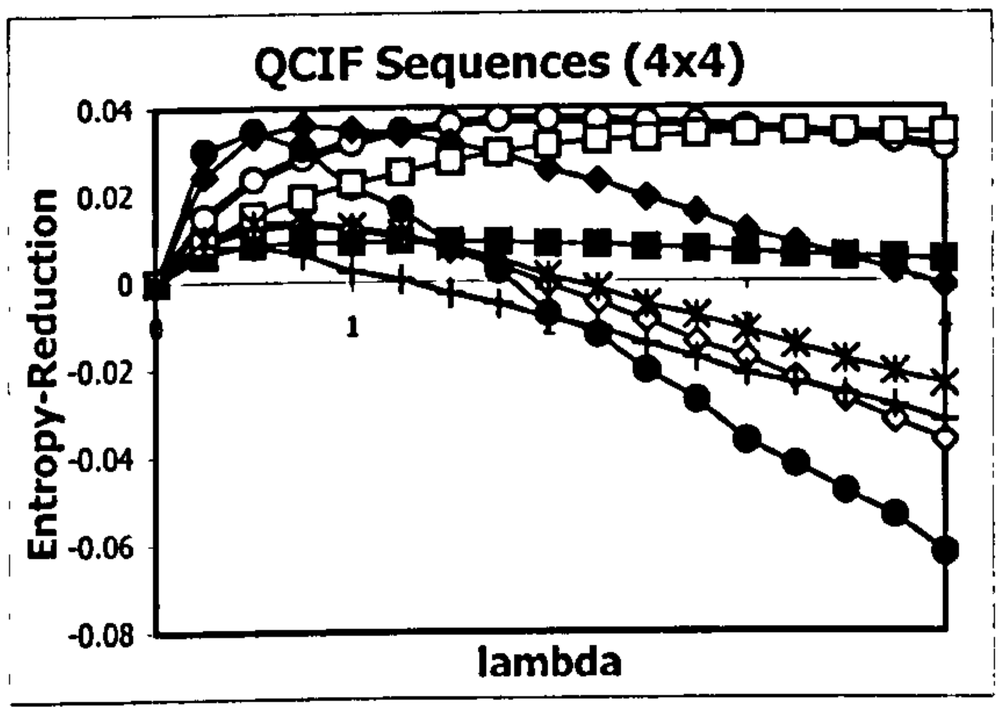
Figure 4.21 An illustration of a typical λ–ΔE_T curves with different block sizes.

4.4.2.5 QBMA Efficiencies for different Test Sequences

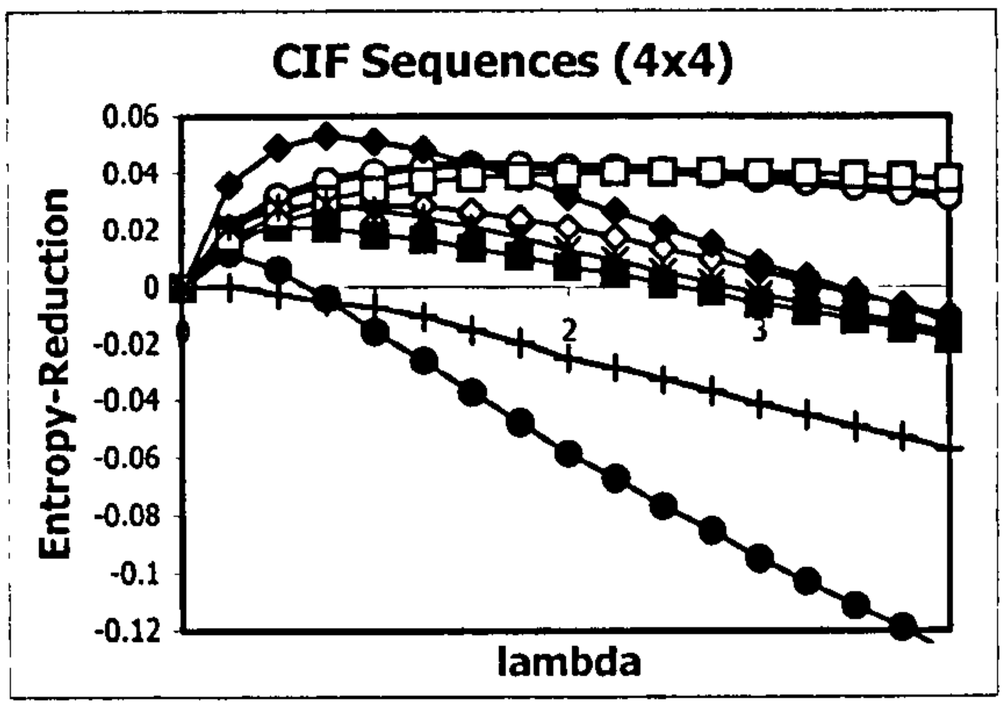
So far we have determined that the QBMA performance on a picture is essentially independent on the size of the picture, and the operating range increases with increasing block size, while peak efficiency reduces as block size increases. The last part of this section explores how $\lambda - \Delta E_T$ curves vary across different test sequences. Figure 4.22 shows six charts, each containing eight $\lambda - \Delta E_T$ curves from the QBMA of the eight sequences. Although each curve within a chart peaks and crosses the x-axis at different values of λ values, there is a common range where all the curves stay positive, indicating that a common operating range can be identified. Hence, even though the value of the smoothness constraint factor (λ) at which QBMA produces optimal result is dependent on the sequence, we can find a sub-optimal value which provides acceptable coding gain for all the sequences. The maximum λ value where each curve remains positive is shown in Table 4.3. The cells with the '>' sign indicates that the positive range is beyond the simulation range. It can be observed that the variation of this range is most markedly across block sizes than across the sequences and picture sizes. By choosing the smallest range for each block size, it is decided that the following smoothness constraint value be used:

- For 4×4 block $\lambda = 0.25$.
- For 8×8 block, $\lambda = 1.0\,\lambda = 1.0$.
- For 16×16 block, $\lambda = 4.0\,\lambda = 4.0$.

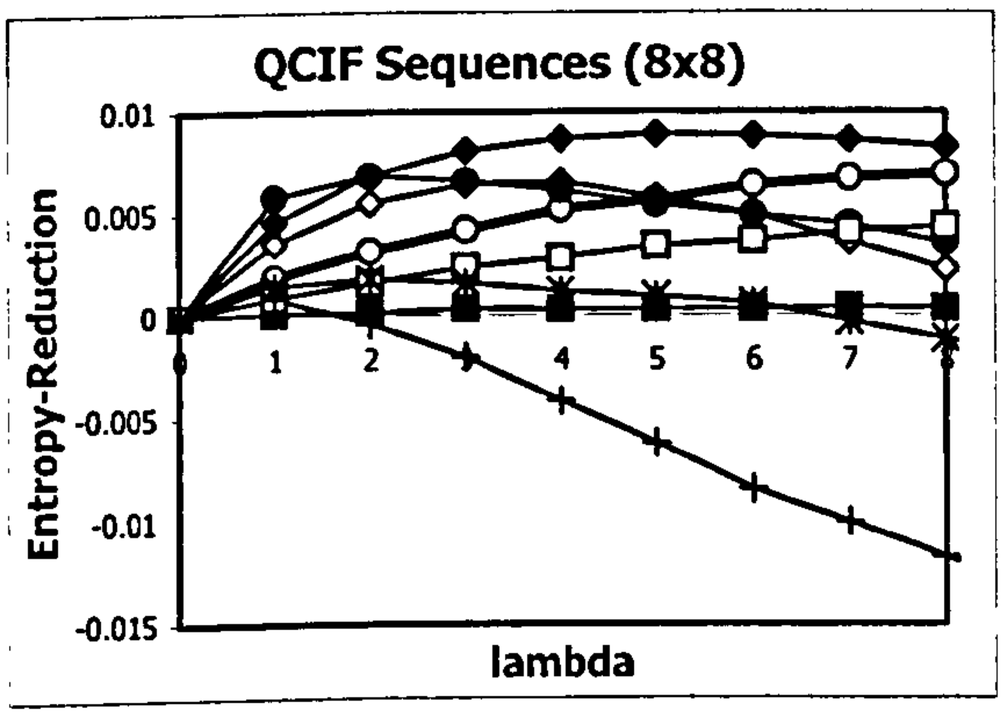
To summarise, QBMA will be used with candidacy ratio $R = 0.1$ and smoothness constraint factor $\lambda = \{1.0, 2.0, 4.0\}$ depending on which block size is used. This setting would offer an improvement in coding efficiency over traditional BMA for any generic natural video sequences.



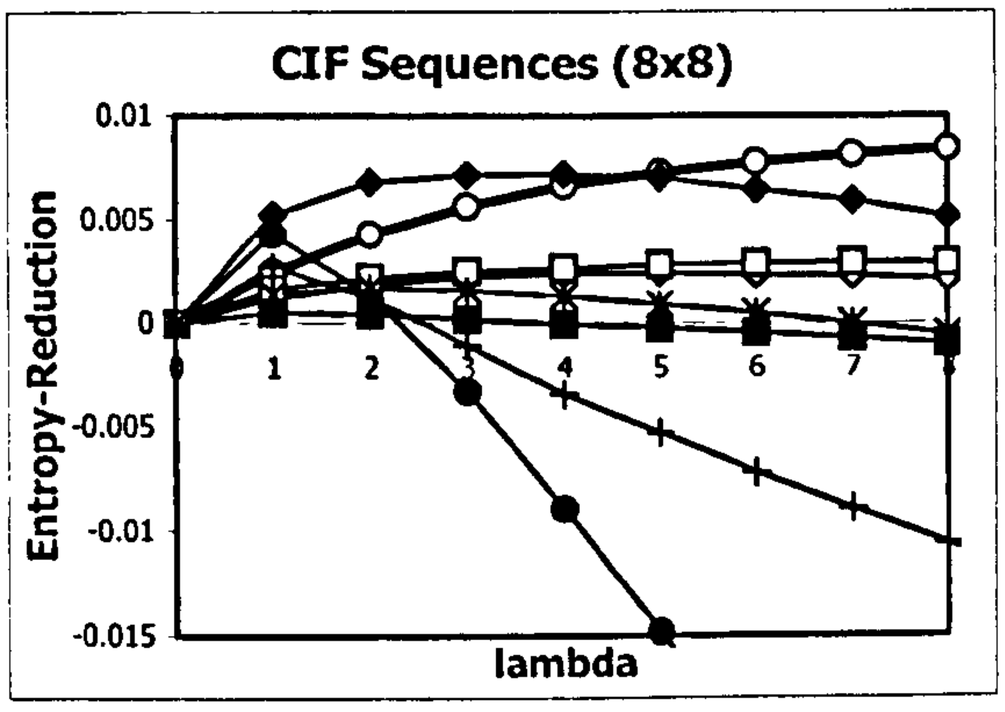
(a)



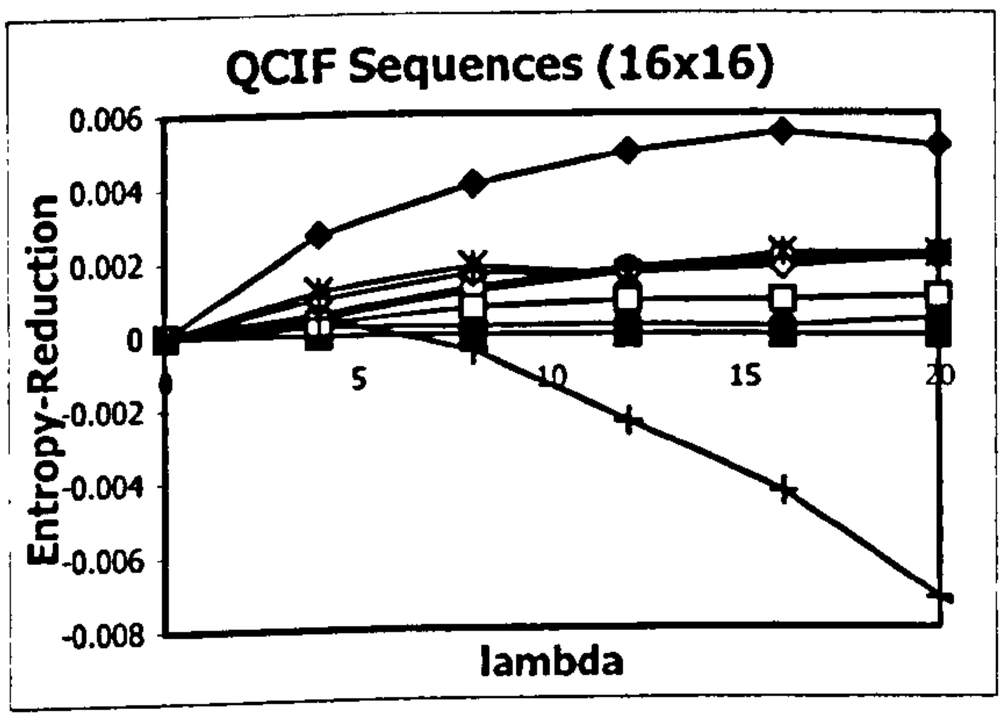
(b)



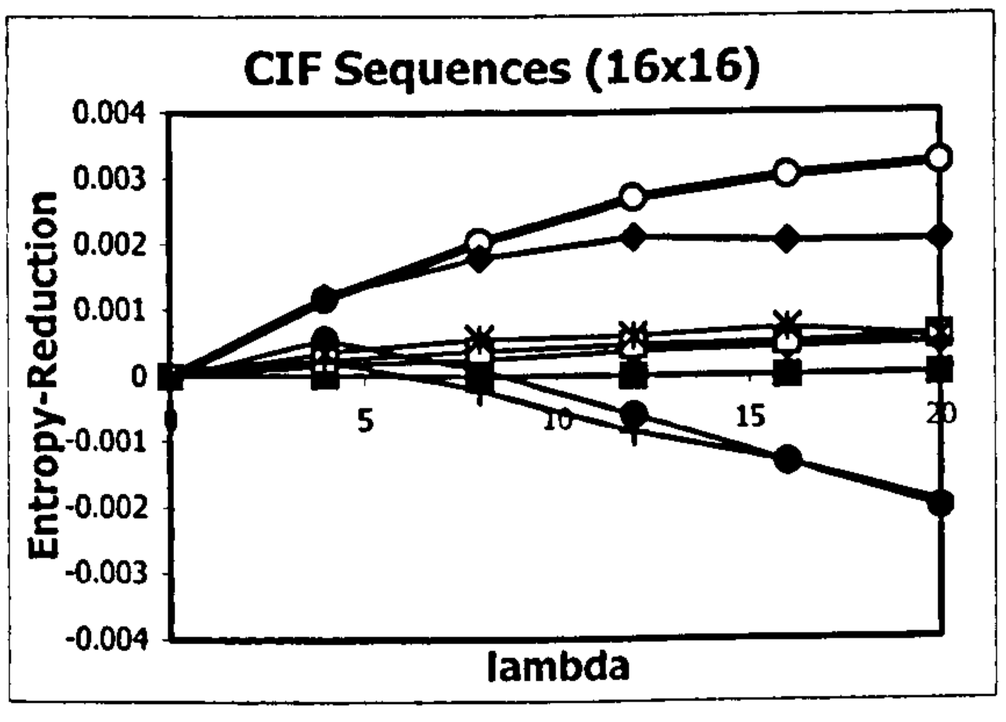
(c)



(d)



(e)



(f)

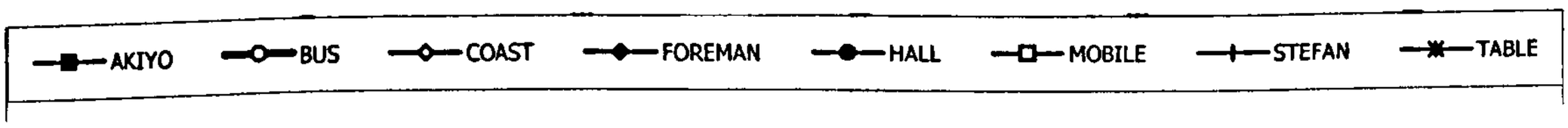


Figure 4.22 Six charts depicting the variation of QBMA efficiency with smoothness constraint factor over different test sequences.

Table 4.3 The λ value beyond which QBMA becomes less efficient then BMA.

Image size	QCIF			CIF		
Block size / Sequence	4×4	8×8	16×16	4×4	8×8	16×16
Akiyo	> 4.0	> 16.0	16.0	2.5	3.0	> 64.0
Bus	> 4.0	> 16.0	> 64.0	> 4.0	> 16.0	> 64.0
Coast	1.75	10.0	> 64.0	3.25	15.0	> 64.0
Foreman	3.75	> 16.0	> 64.0	3.25	14.0	44.0
Hall	1.75	12.0	32.0	0.5	2.0	8.0
Mobile	> 4.0	> 16.0	> 64.0	> 4.0	> 16.0	> 64.0
Stefan	1.0	1.0	4.0	0.25	2.0	4.0

4.4.2.6 Performance of the Finalized QBMA

Finally, the overall performance of QBMA under the preset candidacy ratio and smoothness constraint factor values is compared. Figure 4.23 and Figure 4.24 shows charts comparing the entropies achieved by BMA and QBMA of QCIF and CIF sequences respectively. The top charts show the entropies of the motion vector fields. Throughout all the sequences QBMA produces less motion entropies, due to the smoothness constraint. From middle graphs indicates the increase in residue entropy due to this smoothing. This is natural as the residual obtained from QMBA is not optimal in the SAD sense. The combined entropies are shown in the bottom charts. They show that the total entropies of all the sequences are improved by QBMA.

In terms of processing times, Figure 4.25 shows that for both CIF and QCIF sequences, QBMA does not require much additional processing time compared with that required by full-search BMA.

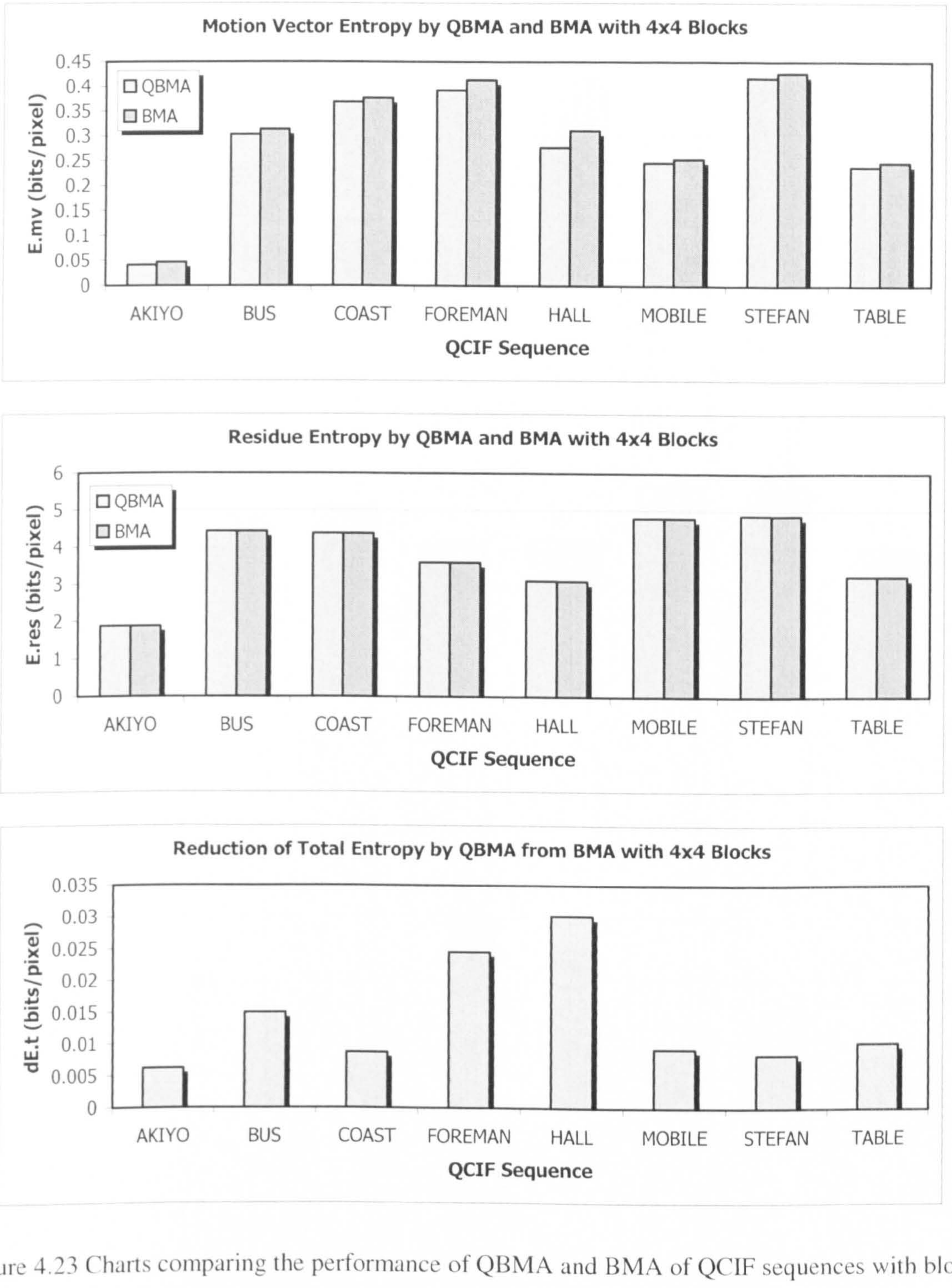
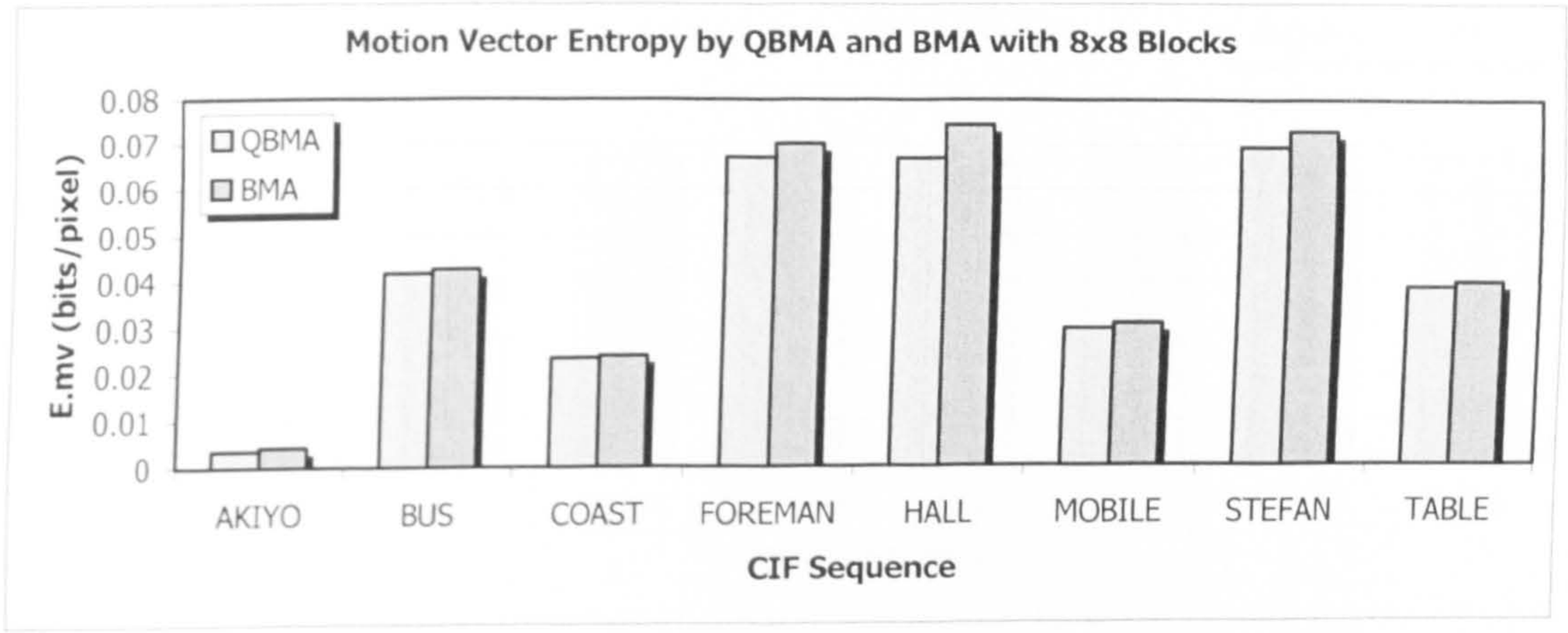
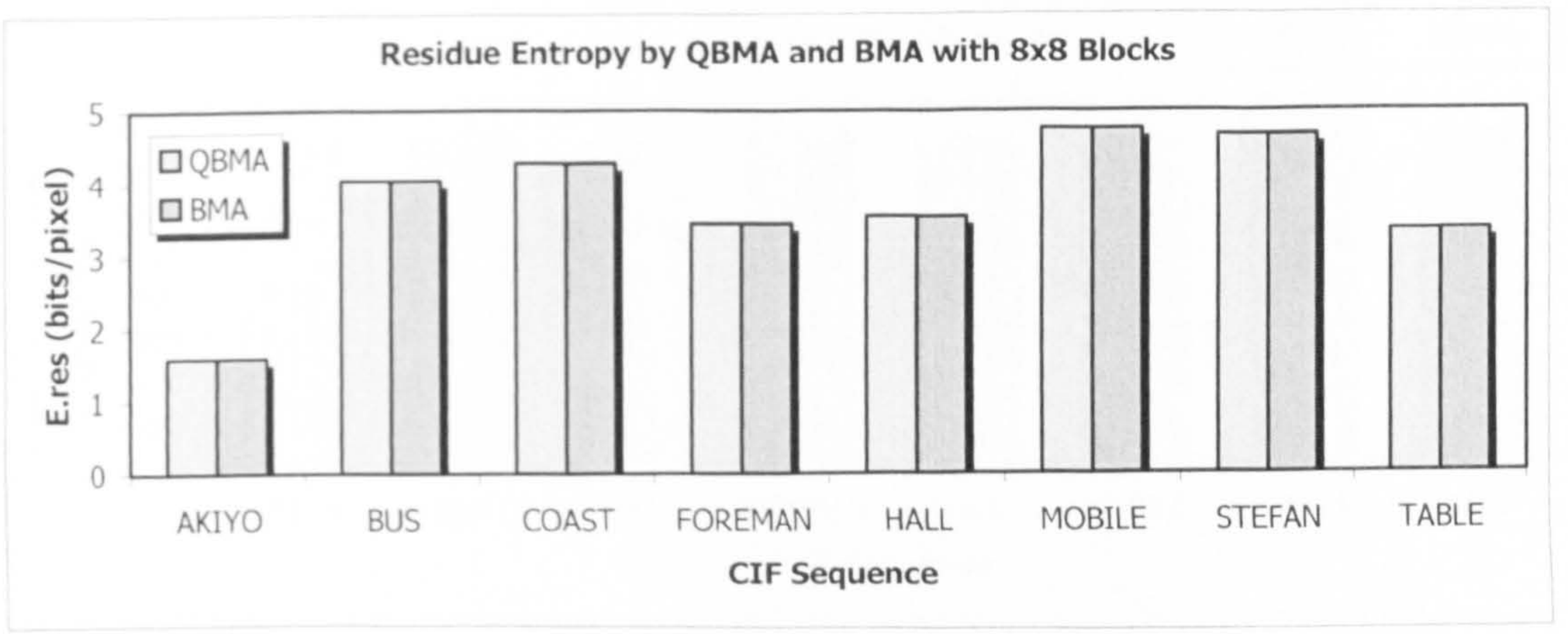


Figure 4.23 Charts comparing the performance of QBMA and BMA of QCIF sequences with block size of 4×4. Top: motion vector entropies; middle: residue entropies; bottom: reduction in total entropies.





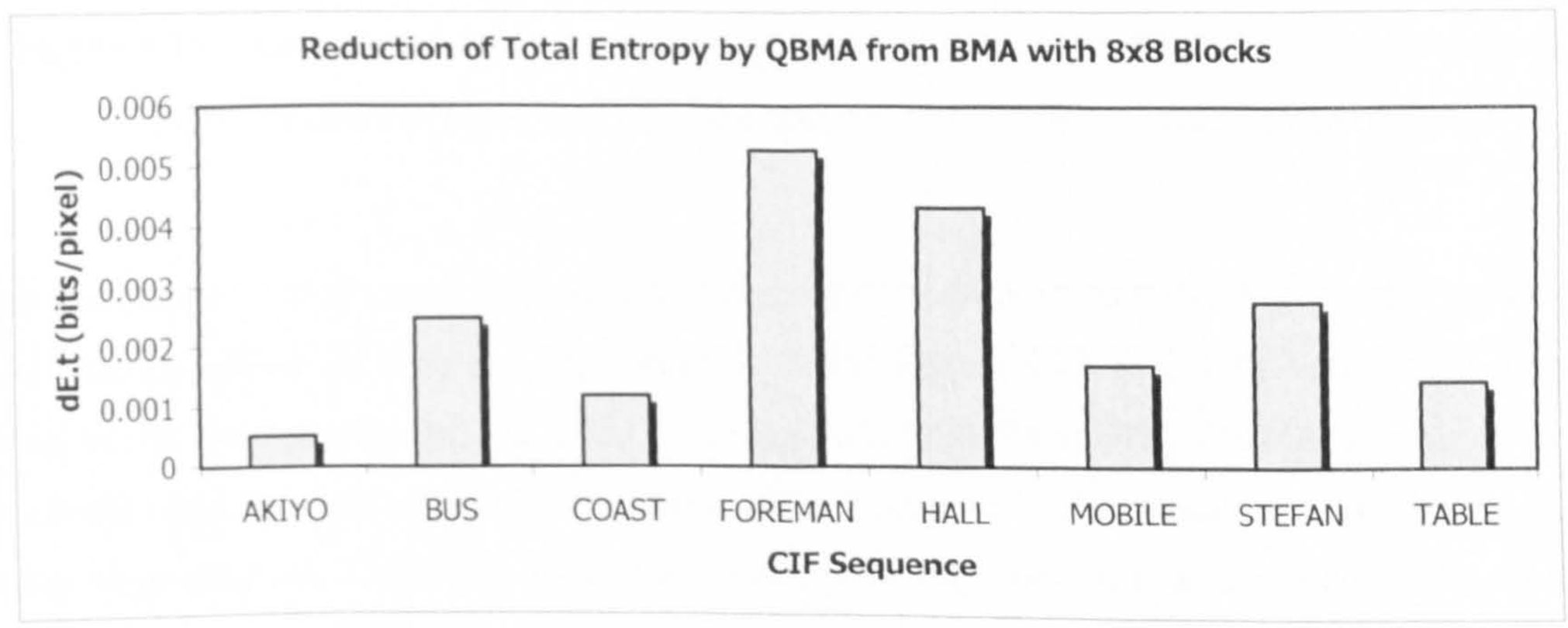
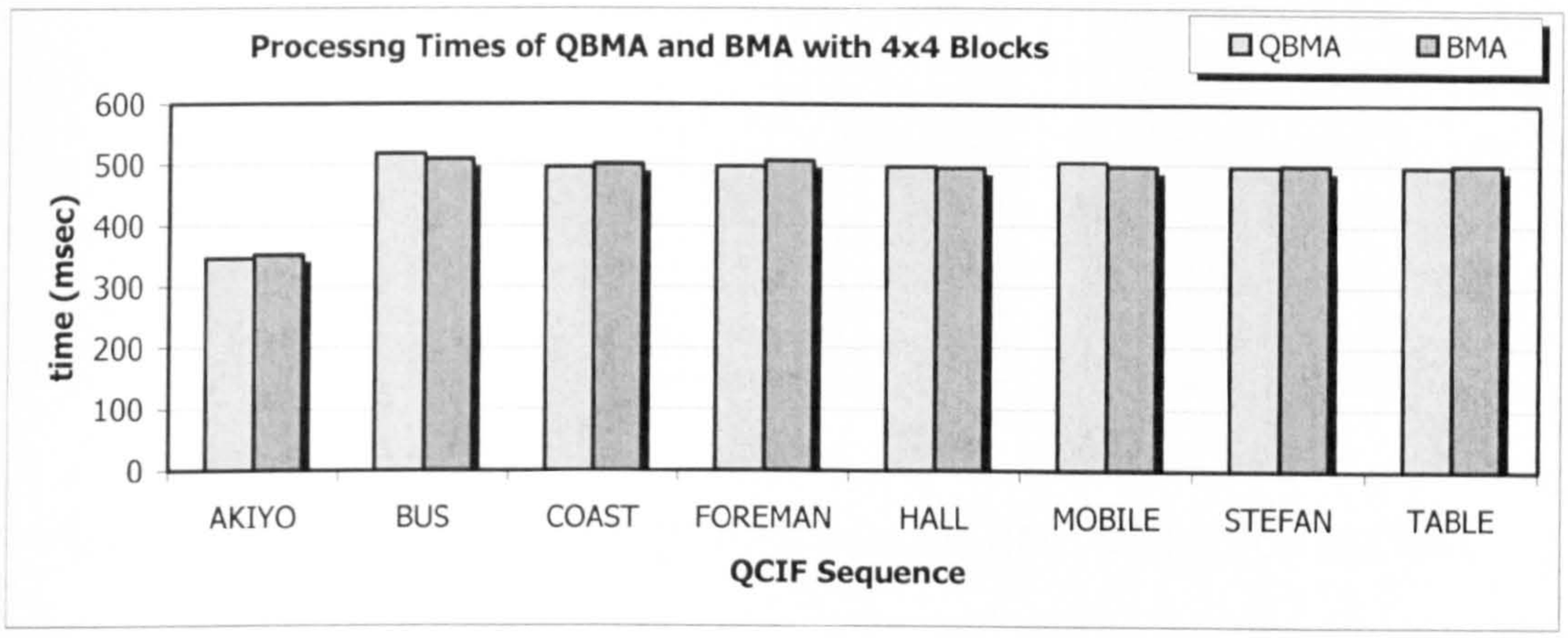


Figure 4.24 Charts comparing the performance of QBMA and BMA of CIF sequences with block size of 8×8. Top: motion vector entropies; middle: residue entropies; bottom: reduction in total entropies.



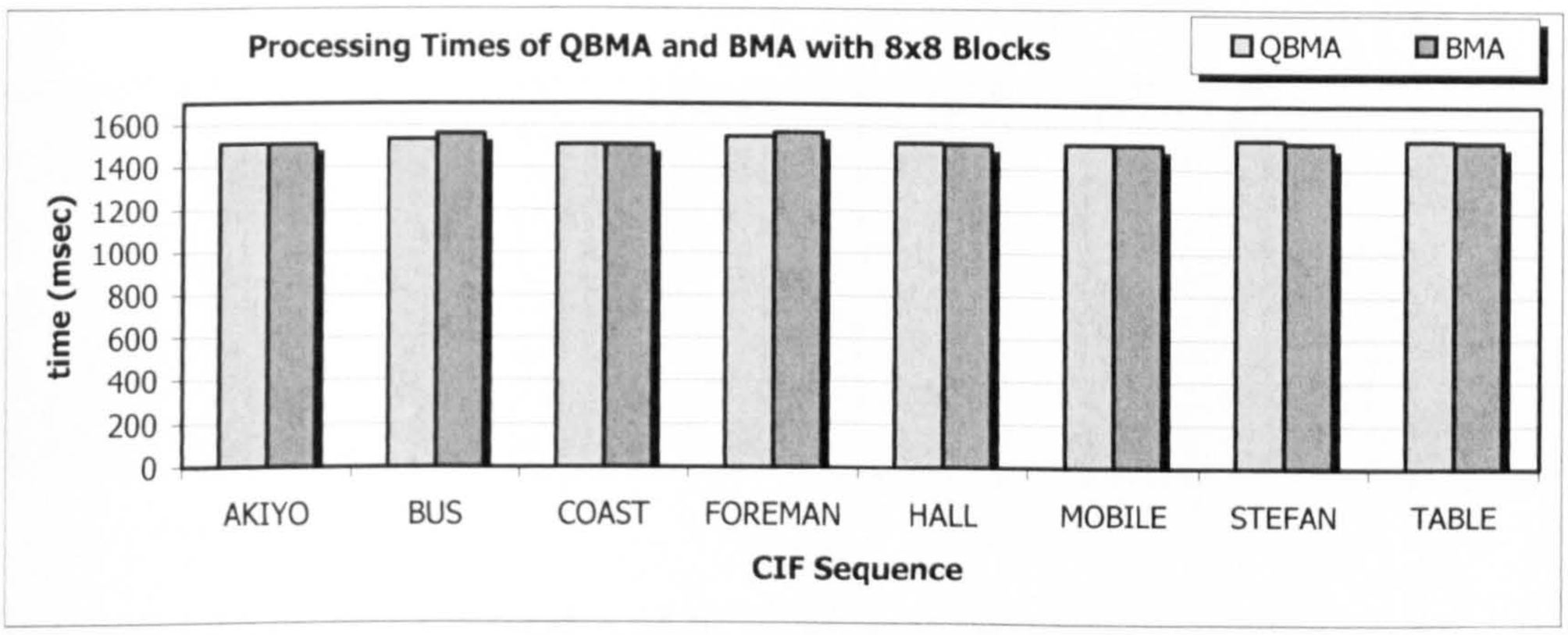
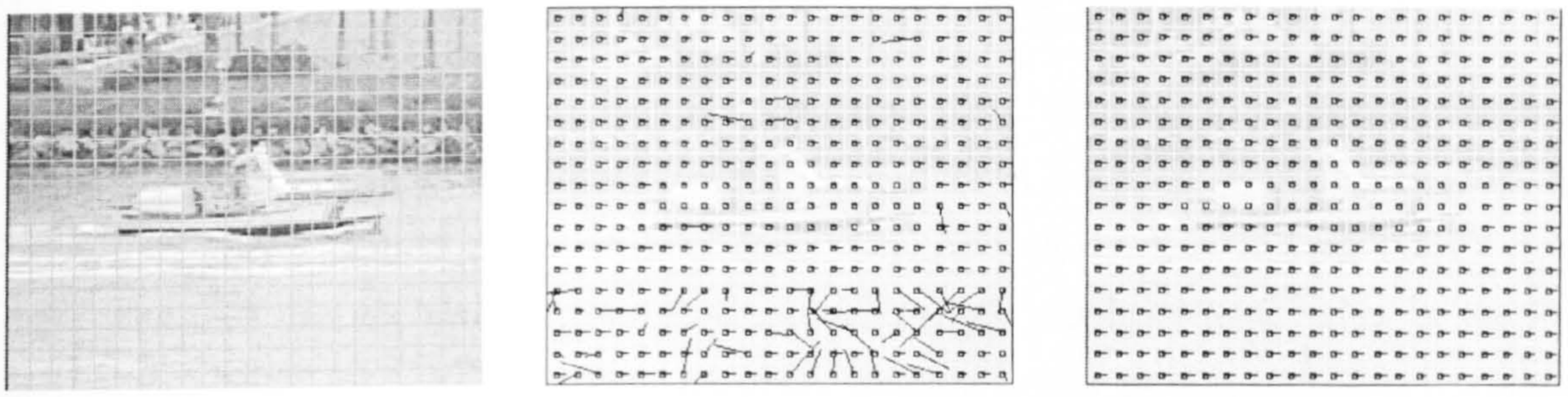


Figure 4.25 Charts comparing the processing times of QBMA and BMA of test sequences. Top: QCIF sequences (block size of 4×4); bottom: CIF sequences (block size of 8×8).

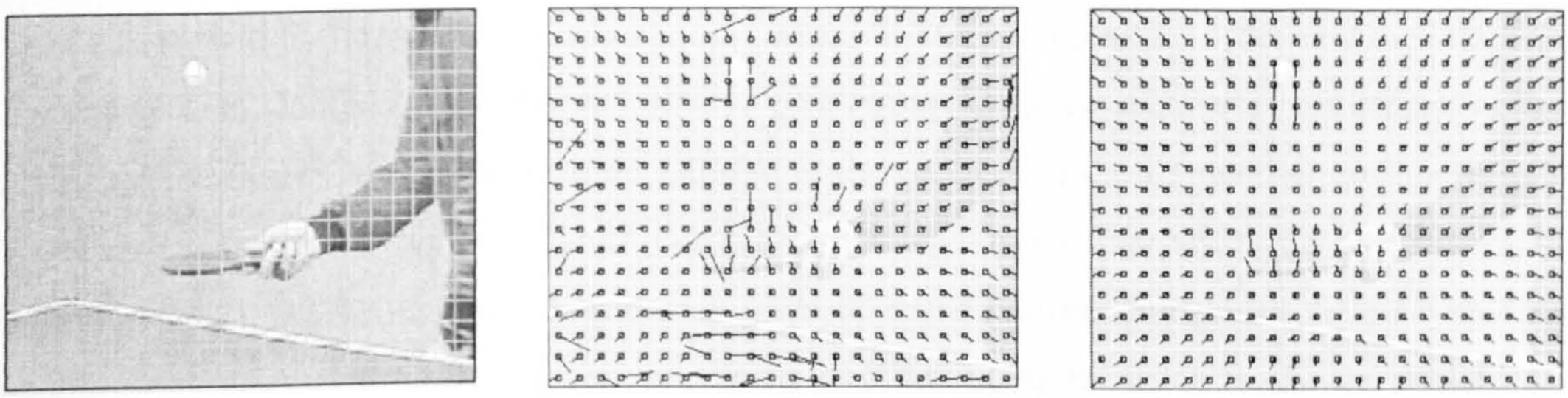
From the perspective of image registration, a comparison of the motion fields generated by the QBMA and BMA are shown in Figure 4.26, Figure 4.27 and Figure 4.28. It can be seen that a much more natural vector field is achieved via QBMA. The motion field obtained by QBMA follows more closely the natural segmentation of the objects in the scene. Hence it is also possible to use the algorithm for motion segmentation. The water waves in coast guard sequence, the background in the table tennis sequence and the floors and walls in the hall sequence all pose serious aperture problem to BMA, which is alleviated by QBMA.

All in all, QBMA can bring about a bitrate reduction of 0.05 bpp for QCIF@10fps sequences and 0.004 bpp for CIF@30fps sequences. This translates to a bit-rate reduction of about 12kbps for both cases.



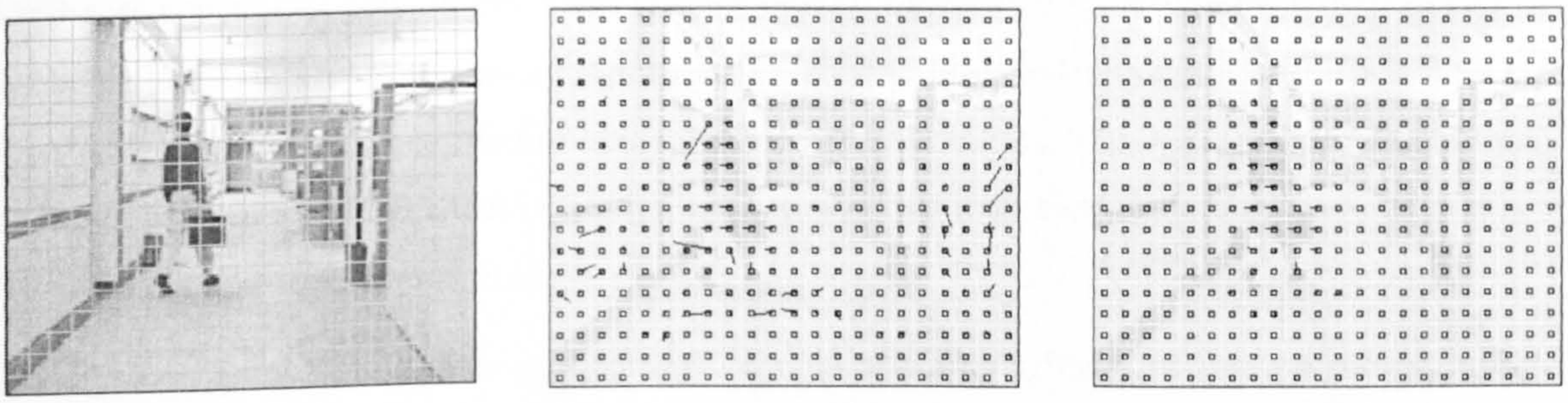
(a) (b) (c)

Figure 4.26. Frame 180 of COAST.QCIF. (a) Input; (b) BMA field; (c) QBMA field.



(a) (b) (c)

Figure 4.27. Frame 33 of TABLE.QCIF. (a) Input; (b) BMA field; (c) QBMA field.



(a) (b) (c)

Figure 4.28. Frame 45 of HALL.QCIF. (a) Input; (b) BMA field; (c) QBMA field.

4.5 Conclusions and Recommendations

This chapter proposed some novel approaches to motion estimation by the block-matching algorithm (BMA). The interpolation-free sub-pixel estimation removes the need to process and store interpolated frames in order to estimate motion vectors to sub-pixel resolutions. Other than for coding purposes,

motion vector fields can also be used as a precursor to global motion estimation and motion segmentation. These applications also benefit from a motion vector field with a higher resolution.

The SAD-map is introduced as an alternative approach to block matching. A reliability measure is introduced which is based on the distribution of the SAD values. A candidate set is identified and their spread is used as an indication of how reliable motion vector from the block is. This reliability measure is more indicative than the measures based on texture, a single minimum SAD value, or the local spatial smoothness of the vector field. This reliability measure is then used in setting up the priority queue in the QBMA algorithm. Other than its use in QBMA, the reliability measure will also be used subsequently as weights for regressive methods for global motion estimation, which will be discussed in later chapters.

The thesis also proposed an alternative approach to BMA, the QBMA, which reduces entropy of the motion vector field by introducing a smoothness constraint and changing the order in which the blocks are processed. In contrast to traditional BMA, processing order is sequential according to raster scanning of the blocks within the picture. QBMA is shown to produce a smoother and more natural motion vector field, and achieve lower combined entropy of motion vectors and residues. The algorithm used in QBMA can be considered a form of statistical relaxation – in classical stochastic relaxation (e.g. simulated annealing and deterministic ICM), motion vector fields are viewed as an integral observation space and the process of finding a stable state involves numerous iterative steps of finding out which configuration produces the lowest 'energy'. The whole process typically requires several seconds or even minutes of processing time on a normal desktop PC. In QBMA, on the other hand, relaxation is done by processing more reliable or 'confident' blocks first, and imposes smoothness constraints only on those less reliable blocks, based on information from their more reliable neighbours. This is a special case of deterministic annealing called Highest Confidence First (HCF) [Mei-97] algorithm does not require any iterative algorithms. As it is a single pass algorithm, the processing time is a several orders of magnitude lower than the other annealing methods and is very well-suited for real-time applications.

In conclusion, the reduction in overall entropy brought about by QBMA makes it a suitable substitute to full-search BMA. As memory and processing requirements for QBMA are not excessively larger then BMA, it is ideal for real-time applications, even in embedded systems. The smoothed field is also a precursor to good block-based motion segmentation algorithms to be discussed in subsequent chapters.

A major area for improvement for the current implementation of QBMA is the choice of smoothness constraint factor. Currently, QBMA offers sub-optimal improvement over BMA because the smoothness constraint factor is constant throughout a particular sequence. Future work involves finding a means of adapting the values to the content and perhaps the factor can be changed from one block to another by analyzing the textural and temporal variation in the neighbourhood of the current block.

Furthermore, QBMA can be used jointly with the sub-pixel models to produce a smooth motion vector field at sub-pixel resolution. Lastly, processing time and memory requirements can be further reduced by means of reducing the resolution of the SAD-map (say, to 2-pixel steps instead of 1-pixel step) and then by applying sub-pixel models to achieve higher resolution.

In the following chapters, various issues global motion estimation and motion segmentations will be investigated and new algorithms based on QBMA and related works will be proposed.

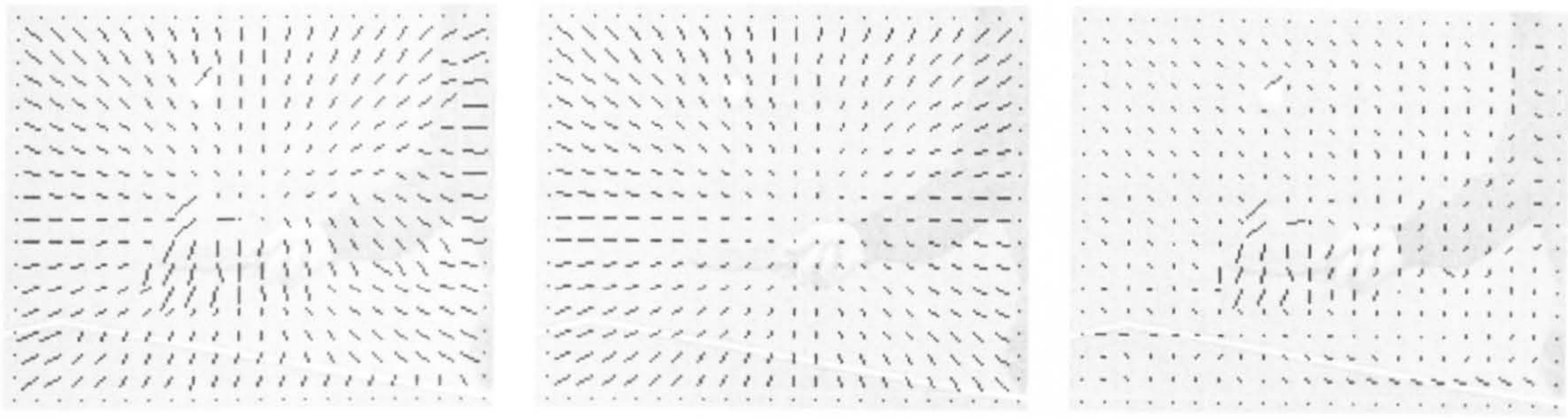
Chapter 5:

Global Motion Estimation

The previous two chapters focused on the acquisition of the motion vector field from two consecutive frames. This and the following chapters are devoted to the means of analysing the field and representing it in a more compact form.

At times, a large part of a motion vector field is attributed to a single motion. In Figure 5.1, the frame from the table tennis sequence with a camera zoom factor of 1.06 creates a substantial number of non-zero vectors (see Figure 5.1 (a)); instead of representing the field as individual vectors as in Figure 5.1 (a), the field can be represented as the sum of a single zoom factor parameter Figure 5.1 (b) and the residual vector field Figure 5.1 (c). This is a much more compact representation of the motion field. As we shall see later, the number of global motion parameters depends on the global motion model. Essentially it ranges from a pure translational model (2 parameters), through the affine model (6 parameters), to the more complicated 12-parameter parametric model. The number of bits required to code these parameters are usually much less than that required encoding the whole field of vectors.

Instinctively, this method of representing the field may not yield significant coding gain when there is only apparent translational motion (induced by camera pan and tilt), as motion vectors of such a field can be differentially coded. However, such fields can still benefit from a 'global' representation. In most cases, the motion vectors are quantized individually and due to a range of noise sources, each vector may be perturbed from its original value and the quantized field is rarely 'uniform'. As an example, consider a scene with a pan-factor of 3.25. When motion vectors are coded to half-pixel resolution (as in H.263), some vectors will be coded as 3.0 and others as 3.5; even with predictive coding, differential vectors of ±0.5 and ±1.0 still exist and this requires extra bits to code. A global representation averages out these perturbations and since only one set of global parameter is required for the whole field, we can afford to code it at a higher resolution, thus producing fewer residues.



(a) (b) (c)

Figure 5.1 An example of global motion: (a) a vector field caused mainly by a camera zoom; (b) vectors caused by the pure zoom factor and (c) the remaining vector after subtracting global motion from the original field. The residual field is much more compact than the original field and takes fewer bits to code.

Detailed scrutiny of (c) in Figure 5.1 reveals that global motion is not entirely eliminated. The errors in the zoom parameter are caused by:

- The vectors component due to local motion, for instance, the table tennis bat.
- The vectors arising from uncovered regions; for instance, the area below the bat, that is absent from the previous frame; motion estimated in this region is meaningless and in most cases unpredictable.
- The inaccuracy of the local motion estimation itself, like the shirt of the player.

Various global motion estimation methods have been proposed to remove such inaccuracies. The remainder of this chapter discusses the principles behind global motion estimation, reviews existing methods and presents a novel algorithm using Hough transform.

5.1 Global Motion Models and Parameters

Global motion (or sometimes referred to as dominant motion) is the apparent motion of a sequence due to the movement of the camera [Par-94]. Consider a point $\mathbf{P} = (X, Y, Z)$ in the scene captured at time t, taking reference from the focal point **O** of the moving camera with the z-axis along the focal axis. Then the corresponding point on the image plane (x, y) at focal length $(z = F)$ would be related to the focal length F by:

$$x = F\frac{X}{Z}, \quad y = F\frac{Y}{Z} \qquad \text{Eq 5-1}$$

The camera motion can be modelled by 2 matrices due to the 3 angles of rotation and translation in three independent directions: in terms of the tilt (α), the pan (β), the rotation along focal axis (γ) and the displacements along x, y and z axes respectively, t_x, t_y, t_z. Camera motion causes a point $\mathbf{P}=(X,Y,Z)$ to be displaced to a new point $\mathbf{P}'=(X',Y',Z')$ with respect to **O**. The relationship is given as [Tek-95]:

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha \\ 0 & -\sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} \cos\beta & 0 & -\sin\beta \\ 0 & 1 & 0 \\ \sin\beta & 0 & \cos\beta \end{bmatrix} \begin{bmatrix} \cos\gamma & \sin\gamma & 0 \\ -\sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \qquad \text{Eq 5-2}$$

$$= \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix}$$

Where

$$\begin{aligned}
r_{11} &= \cos\gamma\cos\beta \\
r_{12} &= \sin\gamma\cos\beta \\
r_{13} &= -\sin\beta \\
r_{21} &= -\sin\gamma\cos\alpha + \cos\gamma\sin\beta\sin\alpha \\
r_{22} &= \cos\gamma\cos\alpha + \sin\gamma\sin\beta\sin\alpha \\
r_{23} &= \cos\beta\sin\alpha \\
r_{31} &= \sin\gamma\sin\alpha + \cos\gamma\sin\beta\cos\alpha \\
r_{32} &= -\cos\gamma\sin\alpha + \sin\gamma\sin\beta\cos\alpha \\
r_{33} &= \cos\beta\cos\alpha \\
t_1 &= t_x \\
t_2 &= t_y \\
t_3 &= t_z
\end{aligned}$$

In addition to the camera motion, the projected image may also be changed because of the variation in focal length, resulting in a zoom factor. Denoting the zoom factor as f, this gives a new focal length fF, and the new projected location becomes:

$$x' = fF\frac{X'}{Z'}, \quad y' = fF\frac{Y'}{Z'} \qquad \text{Eq 5-3}$$

Combining Eq 5-1, Eq 5-2 and Eq 5-3, we form a relationship between (x, y) and (x', y') with respect to the parameters $F, f, \alpha, \beta, \gamma, t_x, t_y$ and t_z as:

$$x' = fF\frac{r_{11}x + r_{12}y + F(r_{13} + t_1/Z)}{r_{31}x + r_{32}y + F(r_{33} + t_3/Z)} \qquad \text{Eq 5-4}$$
$$y' = fF\frac{r_{21}x + r_{22}y + F(r_{23} + t_2/Z)}{r_{31}x + r_{32}y + F(r_{33} + t_3/Z)}$$

By reorganizing parameters in Eq 5-4 we arrive at the widely-used perspective motion model [Kim-99a]:

$$x' = \frac{p_0x + p_1y + p_2}{p_6x + p_7y + 1} \qquad \text{Eq 5-5}$$
$$y' = \frac{p_3x + p_4y + p_5}{p_6x + p_7y + 1}$$

A further assumption that point **P** lies sufficiently far away from the camera reduces the denominators in Eq 5-4 to constants (the x- and y-dependent terms are much smaller than the term with F). This assumption is similar to the orthographical projection, producing the affine motion model:

$$x' = m_0x + m_1y + m_2 \qquad \text{Eq 5-6}$$
$$y' = m_3x + m_4y + m_5$$

A less frequently used model assumes the translational components of the camera motion (t_x, t_y and t_z) are negligible (this is not uncommon in cases where the camera is fixed, as in most surveillance cameras and web-cams used for video conferencing). Coupled with the assumption that the rotational angles are small enough to replace their trigonometric functions with first order approximations, Eq 5-4 can be simplified to:

$$x' = f\left(x + \gamma y - \frac{\beta}{F}x^2 + \frac{\alpha}{F}xy - \beta F\right) \qquad \text{Eq 5-7}$$
$$y' = f\left(-\gamma x + y - \frac{\beta}{F}xy + \frac{\alpha}{F}y^2 + \alpha F\right)$$

With a further assumption that the focal length is sufficiently larger (the case for very narrow-angled images) [Par-94], Eq 5-7 reduces to:

$$x' = f(x + \gamma y - \beta F) = q_0x + q_1y + q_2 \qquad \text{Eq 5-8}$$
$$y' = f(-\gamma x + y + \alpha F) = -q_1x + q_0y + q_3$$

Eq 5-8 is a 4-parameter motion quasi-affine model where q_0 indicates the field divergence, q_1 represents the field curl, q_2 represents the x-displacement and q_3 represents the y-displacement. Further assumptions can reduce Eq 5-8 to a zoom-translational model ($q_1 = 0$) or a pure translational model ($q_0 = q_2 = 0$).

Besides the above, other models have been used which are not directly derived from the 2-dimensional projection of 3-dimensional objects. For example, we can have another 8-parameter bilinear model (Eq 5-9) or the 12-parameter parabolic model (Eq 5-10) [Saw-95]:

$$x' = b_0 + b_1 x + b_2 y + b_3 xy \qquad \text{Eq 5-9}$$
$$y' = b_4 + b_5 x + b_6 y + b_7 xy$$

$$x' = r_0 + r_1 x + r_2 y + r_3 x^2 + r_4 xy + r_5 y^2 \qquad \text{Eq 5-10}$$
$$y' = r_6 + r_7 x + r_8 y + r_9 x^2 + r_{10} xy + r_{11} y^2$$

The common global motion model is summarized in the following table:

Table 5.1 List of global motion parameters.

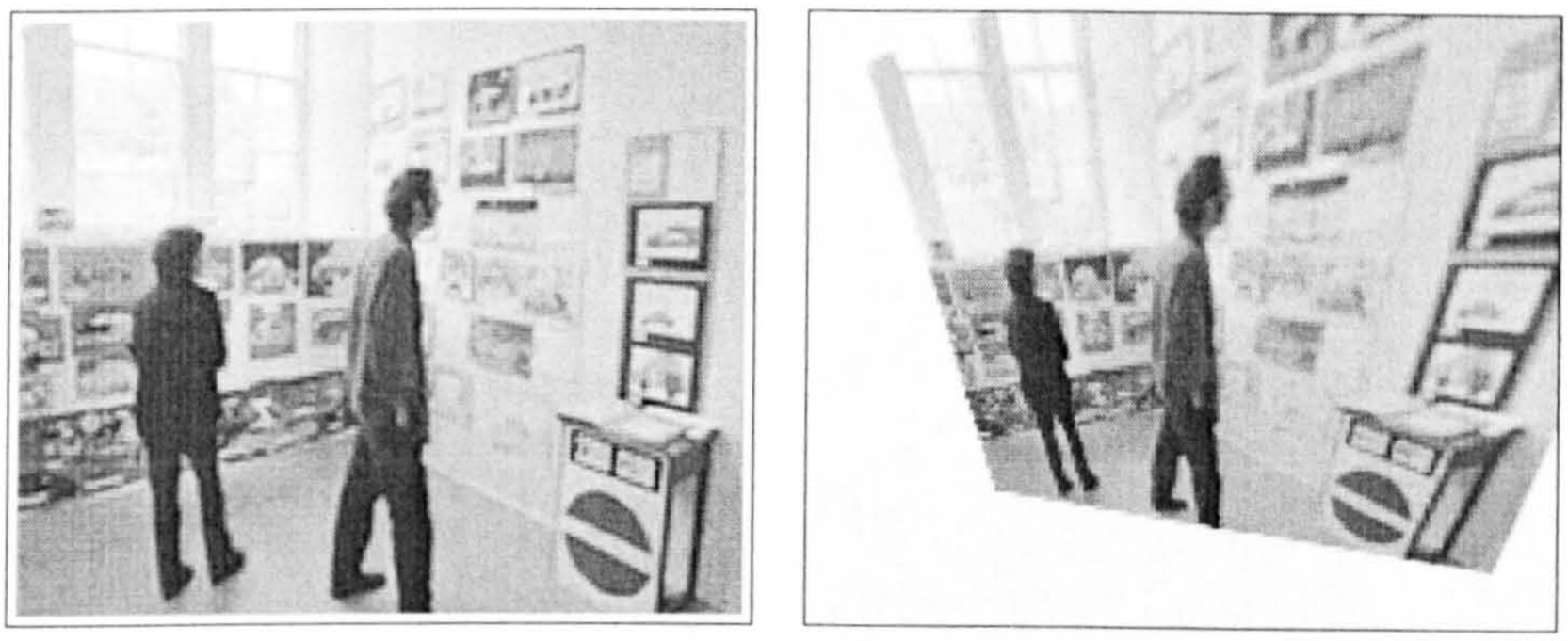
Model Name	Number of parameters	Model Equation
Parabolic	12	$x' = r_0 + r_1 x + r_2 y + r_3 x^2 + r_4 xy + r_5 y^2$ $y' = r_6 + r_7 x + r_8 y + r_9 x^2 + r_{10} xy + r_{11} y^2$
Bilinear	8	$x' = b_0 + b_1 x + b_2 y + b_3 xy$ $y' = b_4 + b_5 x + b_6 y + b_7 xy$
Perspective	8	$x' = \frac{p_0 x + p_1 y + p_2}{p_6 x + p_7 y + 1}$; $y' = \frac{p_3 x + p_4 y + p_5}{p_6 x + p_7 y + 1}$
Affine	6	$x' = m_0 x + m_1 y + m_2$; $y' = m_3 x + m_4 y + m_5$
Quasi-affine	4	$x' = q_0 x + q_1 y + q_2$; $y' = -q_1 x + q_0 y + q_3$
Zoom-Translational	3	$x' = q_0 x + q_2$; $y' = q_0 y + q_3$
Translational	2	$x' = x + q_2$; $y' = y + q_3$

Amongst the available models, the 3-parameter [Eis-91] [Joz-97] [Tse-91] and 4-parameter [Heu-99] [Hil-99] [Nic-91] [Giu-99] [Smo-00a] [Rat-99] models have been the most frequently used in real-time processing in the past. As processors power has increased, the 6-parameter affine model has gained great popularity in recent years [Lin-99] [Zha-98] [Xio-97] [He-01] [Wan-97] [Kel-03] [Smo-00b]

[Ste-99], whereas the 8-parameter perspective model is widely used in offline applications, in cases where non-linear deformation cannot be ignored [Kim-99a]. The bilinear model is useful in coding parameters and is currently used in the H.263+ standard [ITU-98]. The parabolic model [Saw-95] is used for precision global motion estimation for long-term background sprite coding. Other algorithms make use of multiple models with increasing complexity to improve convergence speed [Duf-00] [Smo-99].

5.2 Use of Global motion Estimation

After global motion parameters have been found, it can be applied to numerous applications. As mentioned at the start of this chapter, motion vector field can be made more compact by representing a sparse vector field by the global motion vector itself, plus a residual motion field, which is the difference between the local motion vector and the local vector resulting from the global motion. As seen in Figure 5.1, the residual motion vector field is more uniform and has lower entropy. To provide an insight quantitatively, the combined entropy (residue + motion) in the left panel of Figure 5.1 is 3.61143 bits per pixel (bpp); that in the right panel is 3.4183 bpp. As texture residue is the same in both cases, the difference in entropy is due to that of the motion vectors. Hence, there is a reduction of about 0.2 bpp. However, we have to add the number of bits required to represent the global motion parameters. Using the H.263+ convention of representing global motion vectors, where the parameters are transmitted as 4 motion vectors and assuming a dynamic range of 256 for each component, the global motion information can be estimated at 256*8 = 2048 bits per frame. With a QCIF picture, this is equivalent to 2048/ (176*144) = 0.08 bpp, a minute fraction of the original reduction in entropy. Hence, global motion estimation can reduce bit rates by providing a more compact representation of the original field. Alternatively, GME can be used in Reference Picture Resampling option of H.263+, where the reference picture is "warped' according to the motion parameters. Figure 5.2 illustrate an instance, albeit an extremely exaggerated one, of a warping operating. In warping applications, the purpose of global motion estimation is to find the best parameter such that the warped version of the reference frame is best matched with the current frame.

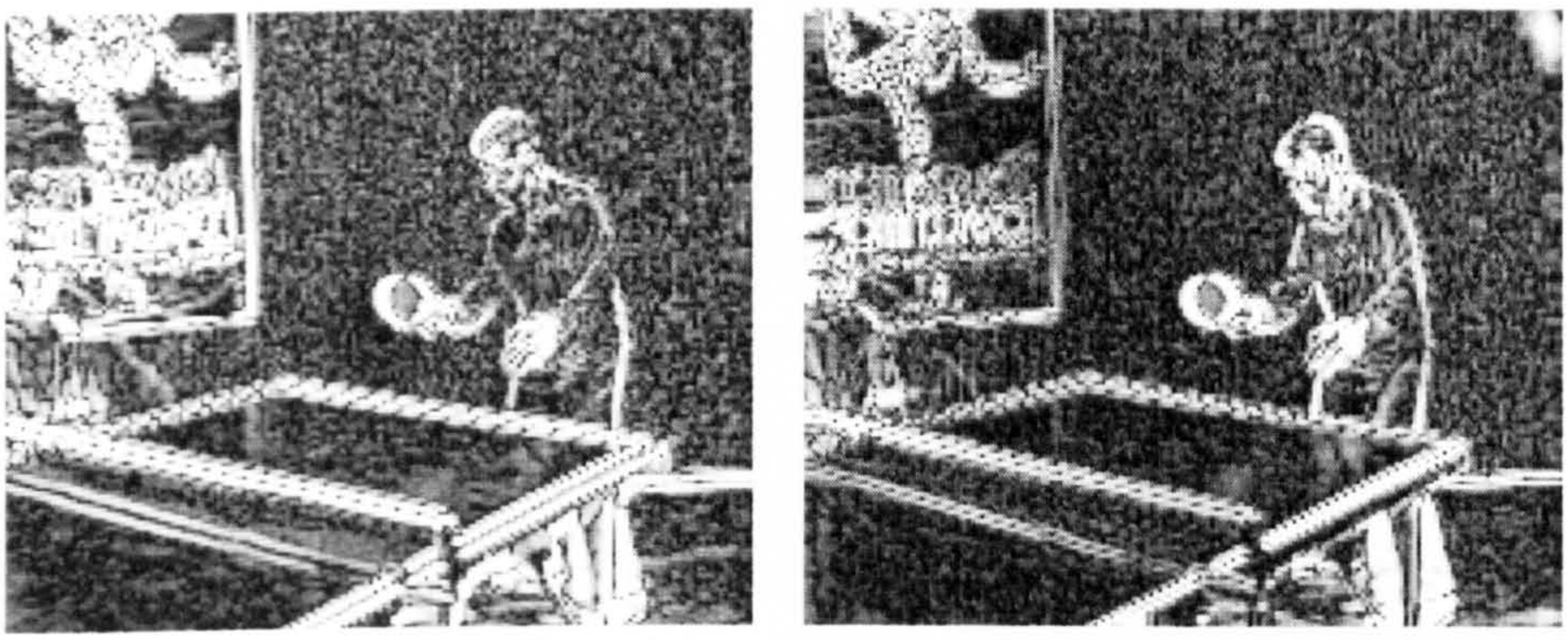


(a) (b)

Figure 5.2 An example of warping. (a) Original image; (b) corresponding warped image according to the perspective model.

There are two issues to be addressed regarding warping operation. Firstly, warped pixels are usually not located on the integer grid; hence sub-pixel interpolation has to be carried out. This is done by using the bilinear interpolation as depicted in Figure 4.5. Secondly, warped pixels may lie beyond the image grid. The pixel is then replaced by the pixel along the picture border nearest to the warped pixel.

The warped reference image is a better match to the input image, as can be seen from the difference images in Figure 5.3. Hence, if instead of performing motion estimation based on the original reference frame, motion estimation can be done using a warped version of the reference frame. As the warped reference is better matched with the input image, the resulting motion estimation will produce less residue entropy, as well as less motion entropy. Close inspection of the two difference frame in Figure 5.3 reveals that most areas in (b) has lower energy than (a) except in moving regions of the foreground objects; hence the use of global motion will be restricted to sequences with small moving objects against a dominant background. It may not be efficient when two or more objects have similar sizes moving independently. However, this may not be an issue if there is a fall back scheme where the non-warped versions can be used when appropriate.



(a) (b)

Figure 5.3 Difference images (a) between current and reference images; (b) between current image and a warped version of the reference. As (b) has lower energy, BMA using the warped reference produces less residue in addition to lower motion vector entropy.

Once global motion has been found, a global motion field can be generated, which is used to compare with the original motion field. Regions containing moving foreground have motion vectors distinctly different from that caused by the global motion. Hence by comparing the local and global motion vectors, moving foregrounds can be extracted, as in Figure 5.4. The foreground/background segmentation has many uses. The foreground can be coded at higher quality or in error resilient applications foreground segments can be more heavily protected by better channel codes. Object segmentation and tracking are also simplified as background is eliminated from the extraction and tracking process.

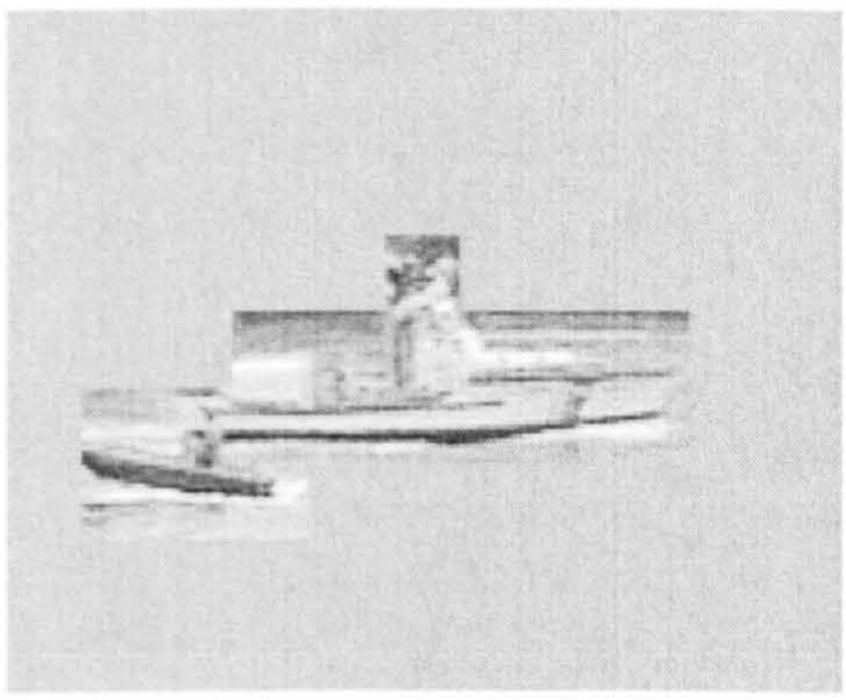


Figure 5.4 Moving foreground extracted from comparing global motion vectors and local motion vectors.

5.3 Existing Global Motion Estimation Techniques

In a more general term, global motion estimation is the process of attributing the apparent difference between two or more frames in a picture sequence to a single motion described by a predefined model. Categorically, there are two approaches to global motion estimation:

- Direct methods
- Indirect methods

Direct methods use spatial gradient within an image and its temporal gradients with respect to another reference image to find a correspondence between the two images. This correspondence is usually evaluated based on the Taylor series expansion of the image space. The indirect methods use motion vectors to estimate the global motion parameters, usually through regression or gradient-descent algorithms. The name 'indirect' is attributed to the fact that a motion vector field is required and it is the field which is used to estimation the motion, not the images themselves. As in the direct methods, both regression and gradient descent algorithms can be used. The following sections discuss the various methods and algorithms in more detail.

5.3.1 Indirect Regression Methods with Motion Vector Fields

When camera motion is the dominant cause of apparent motion between two frames, least-squares methods [Pre-02] can be used to estimate the parameters. Taking the affine model as an illustration, the motion vector $[u, v]$ can be expressed as:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} u \\ v \end{bmatrix} \qquad \text{Eq 5-11}$$

By replacing the parameters $\{m_0, m_1, m_2, m_3, m_4, m_5\}$ in Eq 5-6 by an alternative parameter set $\{a_0, a_1, a_2, a_3, a_4, a_5\}$ with the relationships:

$$\begin{aligned} a_0 &= m_0 - 1; \quad & a_1 &= m_1; \\ a_2 &= m_3; & a_3 &= m_4 - 1; \\ a_4 &= m_2; & a_5 &= m_5 \end{aligned} \qquad \text{Eq 5-12}$$

Eq 5-6 can be than expressed in an alternative form:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} a_0 & a_1 \\ a_2 & a_3 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} a_4 \\ a_5 \end{bmatrix} \qquad \text{Eq 5-13}$$

In Eq 5-13, the $[a_0\ a_1\ a_2\ a_3\ a_4\ a_5]^T$ parameter vector is the model to be evaluated and $\begin{bmatrix} u \\ v \end{bmatrix}$ is the motion vector observed at location $\begin{bmatrix} x \\ y \end{bmatrix}$. By re-arranging terms we arrive at:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} x & y & 0 & 0 & 1 & 0 \\ 0 & 0 & x & y & 0 & 1 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{bmatrix} \qquad \text{Eq 5-14}$$

By cascading $\begin{bmatrix} u_i \\ v_i \end{bmatrix}$-$\begin{bmatrix} x_i \\ y_i \end{bmatrix}$ pairs from a dense or sparse motion vector field, Eq 5-14 is used to form an over-complete equation system:

$$\begin{bmatrix} x_1 & y_1 & 0 & 0 & 1 & 0 \\ 0 & 0 & x_1 & y_1 & 0 & 1 \\ x_1 & y_1 & 0 & 0 & 1 & 0 \\ 0 & 0 & x_2 & y_2 & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{bmatrix} = \begin{bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ \vdots \end{bmatrix} \quad \Rightarrow \quad \mathbf{Pa} = \mathbf{q} \qquad \text{Eq 5-15}$$

The parameter vector $\mathbf{a} = [a_0\ a_1\ a_2\ a_3\ a_4\ a_5]^T$ can be found by means of a least-squares method:

$$\mathbf{a} = (\mathbf{P}^T\mathbf{P})^{-1}\mathbf{P}^T\mathbf{q} \qquad \text{Eq 5-16}$$

Using the above arguments of expressing the affine model in the $\mathbf{Pa} = \mathbf{q}$ form, similar expressions can be obtained for other models. With the exception of the perspective model, the $\mathbf{q}$ vectors are expressed in terms of motion vectors. Table 5.2 shows equation sets for all the models described in Table 5.1 reformatted in the $\mathbf{Pa} = \mathbf{q}$ form, with all parameters represented by $\mathbf{a} = [a_0\ a_1 \cdots]^T$:

Table 5.2 List of global motion estimation equations for least mean square solution.

Model Name	Model Equation
Parabolic	$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} 1 & x & y & x^2 & xy & y^2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & x & y & x^2 & xy & y^2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{11} \end{bmatrix}$
Bilinear	$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} 1 & x & y & xy & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & x & y & xy \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_7 \end{bmatrix}$
Perspective	$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x & y & 1 & 0 & 0 & 0 & xx' & yx' \\ 0 & 0 & 0 & x & y & 1 & xy' & yy' \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_7 \end{bmatrix}$
Affine	$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} x & y & 0 & 0 & 1 & 0 \\ 0 & 0 & x & y & 0 & 1 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_5 \end{bmatrix}$
Quasi-affine	$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} x & y & 1 & 0 \\ -y & x & 0 & 1 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$
Zoom-Translational	$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} x & 1 & 0 \\ y & 0 & 1 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix}$

The solution to Eq 5-16 as used in [Eis-91] and [Rat-99] is very sensitive to outliers. Inaccuracies arise due to motion failure and local motion of moving regions can render the solution meaningless. A common remedy is to apply robust statistics [Pre-02] to reduce the influence of the outliers:

$$\mathbf{a} = (\mathbf{P}^T \mathbf{W} \mathbf{P})^{-1} \mathbf{P}^T \mathbf{W} \mathbf{q} \qquad \text{Eq 5-17}$$

With the weighting matrix $\mathbf{W}$

$$\mathbf{W}=\begin{bmatrix} w_1 & 0 & \cdots & \cdots & 0 \\ 0 & w_1 & & & \vdots \\ \vdots & & \ddots & & \vdots \\ \vdots & & & w_K & 0 \\ 0 & \cdots & \cdots & 0 & w_K \end{bmatrix} \qquad \text{Eq 5-18}$$

In Eq 5-18, K is the cardinality of the motion vector field. Contributions from [Heu-99], [Giu-99] and [Kim-99a] introduce different confidence measures to reduce outliers' effects. Heuer and Kaup [Heu-99] used the variance of the sum-of-difference values found in block matching algorithms (BMA, chapter 3, 4). Giunta and Mascia [Giu-99] employed the concavity of the BMA cost function around the motion vector as a confidence level of the current block. Conversely, Kim and Kim [Kim-99a], on the other hand postulate that the lack of intensity variation within a block reduces the reliability of its motion vector; hence, they used the spatial intensity gradient within the neighbourhood B_k as the k^{th} weight:

$$w_k = \sum_{(x,y)\in B_k} \sqrt{\nabla_x^2 I(x,y,t) + \nabla_y^2 I(x,y,t)} \qquad \text{Eq 5-19}$$

An iterative process can be employed where a subsequent iteration n uses Eq 5-17 to refine $\mathbf{a}^n$ based on the previous global motion field $\mathbf{q}^{n-1}$. The process continues until a convergence limit or a fixed number of iterations are reached. Let ε_k^n denote the estimation error of the k^{th} motion vector in the motion vector field at the n^{th} iteration. Then:

$$\varepsilon_k^n = \left|u_k - \mu_k^n\right| + \left|v_k - v_k^n\right| \qquad \text{Eq 5-20}$$

$$\begin{bmatrix} \mu_1^n \\ v_1^n \\ \vdots \\ \mu_K^n \\ v_K^n \end{bmatrix} = P \begin{bmatrix} a_0^{(n-1)} \\ a_1^{(n-1)} \\ \vdots \end{bmatrix}$$

where w_n is a non-increasing function of ε_n, $w(\varepsilon_n)$, thus reducing the nth observation's contribution to the least square estimation. Various regression methods differ in the choice of function $w(\bullet)$, sometimes referred to as the M-estimator. In [Smo-00a], Smolić et. al. used the Tukey biweight as the M-estimator:

$$w(\varepsilon)=\begin{cases}\left(1-\left(\dfrac{\varepsilon}{cm_\varepsilon}\right)^2\right)^2 & \varepsilon<cm_\varepsilon\\ 0 & \varepsilon>cm_\varepsilon\end{cases} \qquad \text{Eq 5-21}$$

$$m_\varepsilon=\frac{1}{N}\sum_n \varepsilon_n$$

where c is the tuning constant. Xiong, Chiang and Zhang [Xio-97] provided an alternative weight measure based on the intensity difference, weighted with the inverse of spatial gradient:

$$n(\mathbf{p})=\frac{I(\mathbf{p},t)-I(\mathbf{p}-\mathbf{v}(\mathbf{p};0),t)}{\left|\nabla_{\mathbf{p}} I(\mathbf{p},t)\right|} \qquad \text{Eq 5-22}$$

$$\left|\nabla_{\mathbf{p}} I(\mathbf{p},t)\right|=\sqrt{(I_x)^2+(I_y)^2}$$

This weight measure is very simple to implement, but it bears no relation to the actual reliability of the motion vector at the particular location. As will be seen later, this thesis proposes a more relevant measure which provides more relevant weights according to the reliability of the local motion vector.

5.3.2 Indirect Gradient Descent Methods using Motion Vector Field

An alternative means of solving the global motion estimation problem using a motion vector field involves the use of gradient descent methods. Defining $\{\mathbf{p}_k : k=1\cdots K\}$ as the set of observation points with associated observed motion vector $\mathbf{v}_k$, and $\dot{\mathbf{v}}_k$ to be the motion vector at $\mathbf{p}_k$ due to a global motion with model parameters **a** related through a function:

$$\dot{\mathbf{v}}_k = f(\mathbf{p}_k;\mathbf{a}) \quad k=1\cdots K \qquad \text{Eq 5-23}$$

Then global motion estimation can be envisaged as a minimization of the sum of the Euclidean distances between the observed and modelled motion vector fields:

$$\mathbf{a}^* = \arg\min_{\mathbf{a}} E^2(\mathbf{a}) \qquad \text{Eq 5-24}$$

$$E^2(\mathbf{a})=\sum_{i=1}^{N}\left\|\mathbf{v}_k - f(\mathbf{p}_k;\mathbf{a})\right\|^2$$

Solving this equation using least-squares methods leads to the regression method described in the previous section. Alternatively, the problem can also be solved via iterative gradient descent methods.

Similar to pel-recursive methods in local motion estimation (differing only in the dimensionality of the variable to be optimized), we can use the steepest descent method:

$$\mathbf{a}_{n+1} = \mathbf{a}_n - \alpha \nabla_\theta E(\mathbf{a})\big|_{\mathbf{a}_n} \qquad \text{Eq 5-25}$$

$$\nabla_{\mathbf{a}} E(\mathbf{a}) = \begin{bmatrix} \frac{\partial E}{\partial a_1} \\ \frac{\partial E}{\partial a_2} \\ \vdots \\ \frac{\partial E}{\partial a_p} \end{bmatrix}$$

Or the Newton-Raphson method:

$$\mathbf{a}_{n+1} = \mathbf{a}_n - \mathbf{H}^{-1} \nabla_{\mathbf{a}} E(\mathbf{a})\big|_{\mathbf{a}_n} \qquad \text{Eq 5-26}$$

$$\mathbf{H} = \begin{bmatrix} \frac{\partial^2 E}{\partial a_1^2} & \frac{\partial^2 E}{\partial a_1 \partial a_2} & \cdots & \frac{\partial^2 E}{\partial a_1 \partial a_p} \\ \frac{\partial^2 E}{\partial a_2 \partial a_1} & \frac{\partial^2 E}{\partial a_2^2} & \cdots & \frac{\partial^2 E}{\partial a_2 \partial a_p} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 E}{\partial a_p \partial a_1} & \frac{\partial^2 E}{\partial a_p \partial a_1} & \cdots & \frac{\partial^2 E}{\partial a_p^2} \end{bmatrix}$$

The value p in Eq 5-25 and Eq 5-26 is the number of global motion parameters, which depends on the motion model used (see Table 5.1 and Table 5.2). Local motion estimation aims at finding the u and v components of each pixel whereas in global motion estimation, the minimization target is the p parameters in the global motion vector space.

In [Par-94], Park et al use the Levenberg-Marquart method [Pre-02] to solve the problem numerically with a three-parameter model; good estimation results are achieved, although they do not show how it can be used in video coding applications. The global approach proves to be sensitive to outliers due to aperture and occlusion problems. Robust statistics can again be used to reduce the influence of outliers. Kim and Kim [Kim-99b] use outlier rejection ratios to eliminate pixels from the $E(\bullet)$calculations. In addition to introducing robustness, their method reduces the amount of computation per iteration and they obtain good results when their algorithm is incorporated into an H.263 codec.

5.3.3 Gradient Descent with Inter-frame Direct Methods

In direct methods of global motion estimation, motion parameters are evaluated by minimizing the intensity of the image with respect to a warped version of the reference frame. Recalling Eq 3-14 in

pel-recursive methods, the energy term $E(\bullet)$at any point $\mathbf{p}=[x\ y]^T$ is dependent on the motion vector at that point $\mathbf{v}(\mathbf{p},t)$:

$$E(\mathbf{p},t)=[I(\mathbf{p},t)-I(\mathbf{p}-\mathbf{v}(\mathbf{p},t),t-1)]^F \qquad \text{Eq 5-27}$$

For global motion estimation, the motion vector field is a function of position and parameter vector **a**:

$$E(\mathbf{p},t;\mathbf{a})=[I(\mathbf{p},t)-I(\mathbf{p}-\mathbf{v}(\mathbf{p},t;\mathbf{a}),t-1)]^F \qquad \text{Eq 5-28}$$

The form of $\mathbf{v}(\mathbf{p},t;\mathbf{a})$ depends on the motion model as specified in Table 5.2. Global motion estimation then entails the minimization of the total residual energy with respect to **a**:

$$\mathbf{a}^*(t)=\arg\min_{\mathbf{a}}\sum_{\mathbf{p}}[I(\mathbf{p},t)-I(\mathbf{p}-\mathbf{v}(\mathbf{p},t;\mathbf{a}),t-1)]^F \qquad \text{Eq 5-29}$$

In terms of global motion estimation, a more general form of Eq 5-28 and Eq 5-29 is required. The $E(\bullet)$ term has to be expressed in terms of displaced frame difference *DFD* as in Eq 5-30.

$$DFD(\mathbf{p},t;\mathbf{a})=I(\mathbf{p},t)-I(\mathbf{p}-\mathbf{v}(\mathbf{p},t;\mathbf{a}),t-1) \qquad \text{Eq 5-30}$$
$$\mathbf{a}^*(t)=\arg\min_{\mathbf{a}}\sum_{\mathbf{p}}[DFD(\mathbf{p},t;\mathbf{a})]^F$$

In the following discussion, $DFD(\mathbf{p},t;\mathbf{a})$ is referred to the values in the current frame at time t with a specific global motion vector **a**. Hence, the terms t and **a** are dropped from $DFD(\mathbf{p},t;\mathbf{a})$ without the loss of generality. To solve Eq 5-29, Dufaux and Konrad [Duf-00] used a gradient descent method on the 8-parameter perspective model $[a_0\ a_1\ \ldots\ a_7]$ to minimize the residue $\sum DFD(\mathbf{p})$

$$\begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_7 \end{bmatrix}^{(n+1)} = \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_7 \end{bmatrix}^{(n)} - \mathbf{H}^{-1}\mathbf{b} \qquad \text{Eq 5-31}$$

$$\mathbf{H} = \begin{bmatrix} \frac{1}{2}\sum_{\mathbf{p}} \frac{\partial^2 DFD^2(\mathbf{p})}{\partial a_0^2} & \frac{1}{2}\sum_{\mathbf{p}} \frac{\partial^2 DFD^2(\mathbf{p})}{\partial a_1 \partial a_0} & \cdots & \frac{1}{2}\sum_{\mathbf{p}} \frac{\partial^2 DFD^2(\mathbf{p})}{\partial a_0 \partial a_7} \\ \frac{1}{2}\sum_{\mathbf{p}} \frac{\partial^2 DFD^2(\mathbf{p})}{\partial a_0 \partial a_1} & \frac{1}{2}\sum_{\mathbf{p}} \frac{\partial^2 DFD^2(\mathbf{p})}{\partial a_1^2} & & \\ \vdots & & \ddots & \\ \frac{1}{2}\sum_{\mathbf{p}} \frac{\partial^2 DFD^2(\mathbf{p})}{\partial a_0 \partial a_7} & & & \frac{1}{2}\sum_{\mathbf{p}} \frac{\partial^2 DFD^2(\mathbf{p})}{\partial a_7^2} \end{bmatrix}$$

$$\mathbf{b} = \begin{bmatrix} \frac{1}{2}\sum_{\mathbf{p}} \frac{\partial DFD^2(\mathbf{p})}{\partial a_0} \\ \frac{1}{2}\sum_{\mathbf{p}} \frac{\partial DFD^2(\mathbf{p})}{\partial a_1} \\ \vdots \\ \frac{1}{2}\sum_{\mathbf{p}} \frac{\partial DFD^2(\mathbf{p})}{\partial a_7} \end{bmatrix}$$

The second-order terms in **H** and **b** are computationally intractable. By ignoring higher-order terms, they can be approximated by their first derivatives:

$$H_{mn} = \frac{1}{2}\sum_{\mathbf{p}} \frac{\partial^2 DFD^2(\mathbf{p})}{\partial a_m \partial a_n} = \frac{1}{2}\sum_{\mathbf{p}} \frac{\partial DFD(\mathbf{p})}{\partial a_m} \frac{\partial DFD(\mathbf{p})}{\partial a_n} \qquad \text{Eq 5-32}$$

$$b_m = \frac{1}{2}\sum_{\mathbf{p}} \frac{\partial DFD^2(\mathbf{p})}{\partial a_m} = \sum_{\mathbf{p}} DFD(\mathbf{p}) \frac{\partial DFD(\mathbf{p})}{\partial a_m}$$

Instead of using the squared errors as the cost to minimized, robust statistics can be used to reduce the effects of outliers in gradient descent methods by means of M-estimators. By replacing $DFD(\mathbf{p})$ with a simple truncated quadratic M-estimator $\rho(DFD)$ with a fixed threshold t, Dufaux and Konrad managed to improve the convergence rate:

$$\rho(DFD) = \begin{cases} DFD^2 & |DFD| \le t \\ 0 & |DFD| > t \end{cases} \qquad \text{Eq 5-33}$$

Smolić and Ohm [Xio-97] used a similar robust estimation with a regional adaptive threshold:

$$\rho(DFD) = \begin{cases} DFD^2 & (DFD)^2 \le cE \\ 0 & (DFD)^2 > cE \end{cases} \qquad \text{Eq 5-34}$$

$$E = \sum_{\mathbf{p} \in R} DFD(\mathbf{p})$$

To further reduce computational load and the adverse effects of untextured regions, pixels with spatial gradient less than a threshold are removed from the minimization process.

The use of M-estimators is also introduced in [He-01] [Saw-95]. Yuwen He et al [He-01] conducted an extensive survey on the performance of using Eq 5-32 on direct methods of global motion estimation in the MPEG-4 framework.

$$\rho(DFD) = \frac{1}{\sigma^2 + DFD^2} \qquad \text{Eq 5-35}$$

$$\sigma = 1.253\, E(DFD)$$

5.3.4 Regression with Inter-frame Direct Methods

Direct methods can also be approached using the data conservation principles

$$\begin{bmatrix} I_x(x,y) & I_y(x,y) \end{bmatrix} \begin{bmatrix} u(x,y) \\ v(x,y) \end{bmatrix} = \dot{I}(x,y) \qquad \text{Eq 5-36}$$

Where $I_x(x,y)$, $I_y(x,y)$ and $\dot{I}(x,y)$ are the spatial and temporal gradient, and $u(x,y), v(x,y)$ are the motion vector components. Taking the affine model to replace the motion vector components, Eq 5-36 is reformulated by:

$$\begin{bmatrix} xI_x(x,y) & yI_x(x,y) & xI_y(x,y) & yI_y(x,y) & I_x(x,y) & I_y(x,y) \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{bmatrix} = \dot{I}(x,y) \qquad \text{Eq 5-37}$$

A over-complete equation system can then be obtained by cascading all the observation sets $\{x, y, I_x(x,y), I_y(x,y), i(x,y)\}$:

$$\mathbf{\Pi a} = \mathbf{\theta} \qquad \text{Eq 5-38}$$

$$\mathbf{\Pi} = \begin{bmatrix} x_1 I_{x,1} & y_1 I_{x,1} & x_1 I_{y,1} & y_1 I_{y,1} & I_{x,1} & I_{y,1} \\ x_2 I_{x,2} & y_2 I_{x,2} & x_2 I_{y,2} & y_2 I_{y,2} & I_{x,2} & I_{y,2} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \quad \mathbf{a} = \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{bmatrix} \quad \mathbf{0} = \begin{bmatrix} I_1 \\ I_2 \\ \vdots \end{bmatrix}$$

Regression is then used to find the value of **a** which minimize the squared errors:

$$\mathbf{\Pi a} = \mathbf{\theta} \quad \Rightarrow \quad \mathbf{a} = \left(\mathbf{\Pi}^T \mathbf{\Pi}\right)^{-1} \mathbf{\Pi}^T \mathbf{\theta} \qquad \text{Eq 5-39}$$

As in the gradient descent methods, some points may be more reliable than others; hence a weight can be added to each point in the form of weights $\{w_i\}$:

$$\mathbf{\Xi\Pi a} = \mathbf{\Xi 0} \quad \Rightarrow \quad \mathbf{a} = \left(\mathbf{\Pi}^T \mathbf{\Xi}^T \mathbf{\Xi\Pi}\right)^{-1} \mathbf{\Pi}^T \mathbf{\Xi}^T \mathbf{\Xi 0} \qquad \text{Eq 5-40}$$

$$\mathbf{a} = \left(\mathbf{\Pi}^T \mathbf{W\Pi}\right)^{-1} \mathbf{\Pi}^T \mathbf{W0}$$

$$\mathbf{W} = \mathbf{\Xi}^T \mathbf{\Xi} = \begin{bmatrix} w_1 & 0 & 0 \\ 0 & w_2 & 0 \\ 0 & 0 & \ddots \end{bmatrix}$$

5.3.5 Robust Statistics

Before proceeding to the next section, the use of robust statistics will be addressed. In both gradient-based and regression approaches, the basic rule is to minimize the cost measured as a squared error of either deviation of motion vectors (in indirect methods) or difference in pixel data (in direct methods). For simplicity and sometimes computational tractability, the basic assumption is that this error is Gaussian distributed. As a result the least squares algorithms are very sensitive to a large errors value. A single observation with a large error (called an outlier) can cause the estimated parameters to deviate significantly from its true value. An illustration is shown in Figure 5.5:

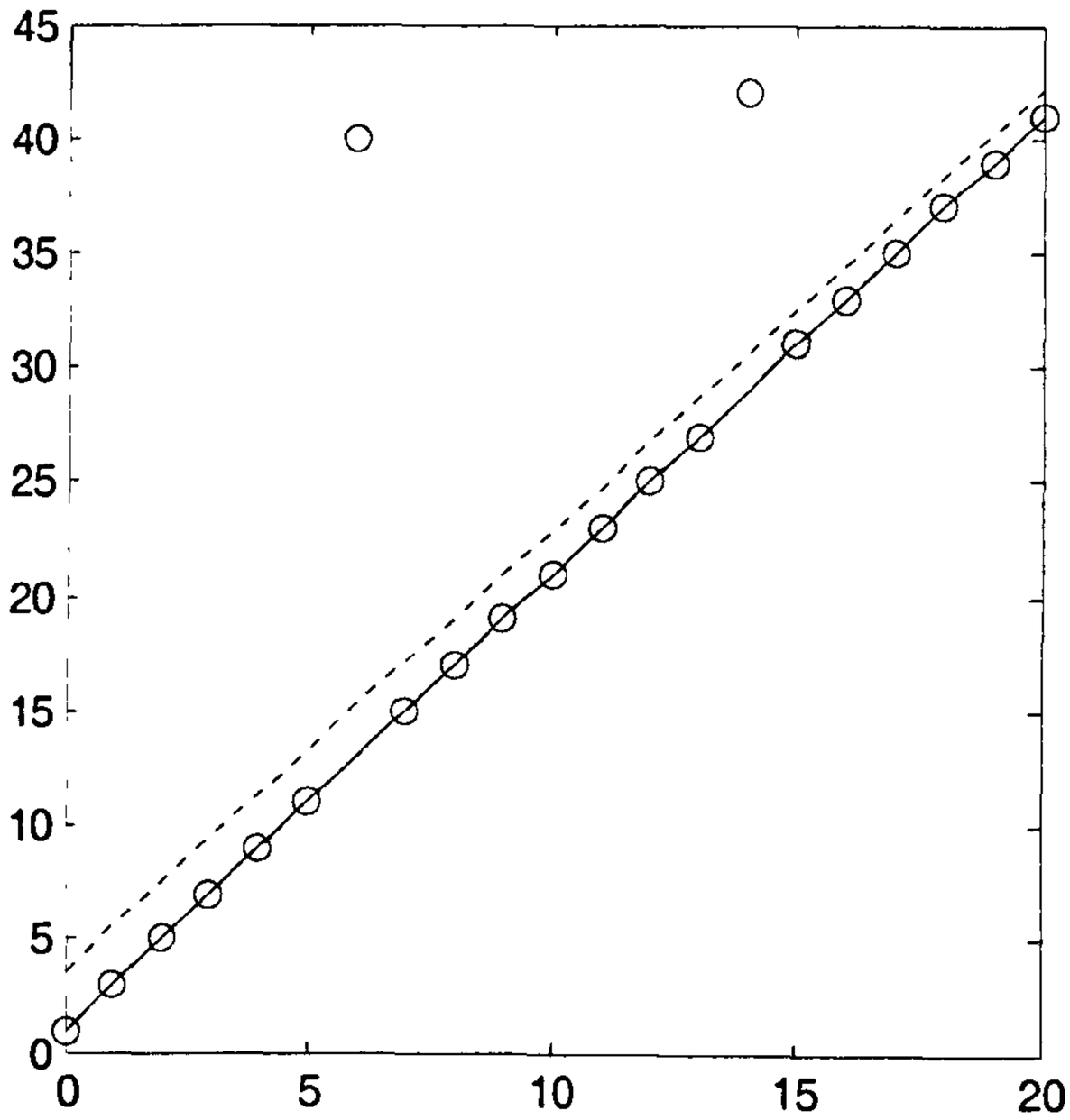


Figure 5.5. Illustration of using robust statistics for linear regression in line fitting application. Point set contains two outliers. Dotted line is a result of linear regression; solid line is found from single iteration of regression using the Tukey's biweight M-estimator

The observation set is perturbed by two outliers. The two outliers cause linear regression (see Eq 5-16) to produce an estimated line (dotted line) which deviates from the original equation (solid line). By using robust statistics with weights of the Tukey's biweight M-estimator, Eq 5-17 produces the solid line, which coincides with the actual line. In global motion estimation, robust statistics is very crucial due to the presence of various sources of noise, like moving object boundaries, change in illumination and occlusions. Ordinary least squares method is not robust because the objective function $E(t)$ (where t is the parameter to be estimated)

$$E(t) = \sum_{k=1}^{K} e_k^2(t) \qquad \text{Eq 5-41}$$

increases infinitely when a single error value $e_k^2(t)$ grows. Hence, a single outlier is sufficient to divert the estimating process significantly. It is important to restrict the effect of outliers by replacing the square function by some function $\rho(e_k)$ which increases initially with e_k and decreases rapidly. This is

the same function described in Eq 5-32 and Eq 5-33. This function is called the M-estimator. Various versions of M-estimators have been proposed in past literature and are usually described by the derivative $\psi(r)=\frac{d}{dr}\rho(r)$. Common M-estimators are described in Table 5.3. The third column, $w(r)$, is the version used as weights in linear regression like Eq 5-40. The measure in Table 5.3, the L_2 estimate, is used in the basic MSE-based regression. In terms of complexity, the Andrew's and Welsch estimates are not ideal for real-time and hardware implementation as trigonometric and exponential functions are used. The simpler L_1, Huber and Tukey's biweight estimates were found not to provide the necessary robustness towards motion outliers in regression-based GME; the Cauchy estimate has been shown to be the most suitable estimate in GME application. The scaling factor c in the Cauchy estimation function is used to normalize the measure with respect to the spread of the estimate. A few spread measures were tried (median of absolute deviation, variance and range) and variance was found to be most suitable measure of c. In the following discussion, the Cauchy estimate with variance as the scale factor is used.

Table 5.3 List of M-estimators for robust statistics.

Type	$\psi(r)$	$\rho(r)$	$w(r)$
L_2	r	$\frac{1}{2}r^2$	1
L_1	$\text{sgn}(r)$	$\lvert r\rvert$	$\frac{\text{sgn}(r)}{r}$
Huber	$\begin{cases} r & \lvert r\rvert \le k \\ k\,\text{sgn}(r) & \lvert r\rvert > k \end{cases}$	$\begin{cases} \frac{1}{2}r^2 & \lvert r\rvert \le k \\ k\lvert r\rvert - \frac{1}{2}k^2 & \lvert r\rvert > k \end{cases}$	$\begin{cases} 1 & \lvert r\rvert \le k \\ \frac{k\,\text{sgn}(r)}{r} & \lvert r\rvert > k \end{cases}$
Cauchy	$\frac{r}{1+\left(\frac{r}{c}\right)^2}$	$\frac{c^2}{2}\log\left[1+\left(\frac{r}{c}\right)^2\right]$	$\frac{1}{1+\left(\frac{r}{c}\right)^2}$
Tukey's biweight	$\begin{cases} r\left(1-r^2\right)^2 & \lvert r\rvert \le 1 \\ 0 & \lvert r\rvert > 1 \end{cases}$	$\begin{cases} \frac{1}{6}\left[1-\left(1-r^2\right)^3\right] & \lvert r\rvert \le 1 \\ \frac{1}{6} & \lvert r\rvert > 1 \end{cases}$	$\begin{cases} \left(1-r^2\right)^2 & \lvert r\rvert \le 1 \\ 0 & \lvert r\rvert > 1 \end{cases}$
Andrew's	$\begin{cases} \frac{1}{\pi}\sin \pi r & \lvert r\rvert \le 1 \\ 0 & \lvert r\rvert > 1 \end{cases}$	$\begin{cases} \frac{1}{\pi^2}\left(1-\cos \pi r\right) & \lvert r\rvert \le 1 \\ \frac{2}{\pi^2} & \lvert r\rvert > 1 \end{cases}$	$\begin{cases} \frac{1}{\pi r}\sin \pi r & \lvert r\rvert \le 1 \\ 0 & \lvert r\rvert > 1 \end{cases}$
Welsch	$r\exp\left(-\left(\frac{r}{c}\right)^2\right)$	$\frac{c^2}{2}\left[1-\exp\left(-\left(\frac{r}{c}\right)^2\right)\right]$	$\exp\left(-\left(\frac{r}{c}\right)^2\right)$

5.4 SAD-based Iterative Regression for GME (SIRGME)

Of the four iterative methods discussed in the previous section, regression using motion vector flow is the least time consuming. Especially when a sparse motion vector field is used, processing times are substantially reduced compared with the gradient-based algorithms with both direct and indirect methods. The previous section has laid down the basic algorithm of Iterative Regression for GME (IRGME). Based on the use of SAD-map and its related techniques, a novel improved version of IRGME is proposed which gives a better estimation potential. This is termed SAD-map-based IRGME (SIRGME). First, the affine model $\{a_0, a_1, a_2, a_3, a_4, a_5\}$ is adopted due to its linearity properties and its

ease of implementation. It also provides a good estimation to the sequences which we used in the simulation.

For each block k with centres $\mathbf{p}_k = (x_k, y_k)$ measured in pixels from the centre of the picture, let $\mathbf{v}_k = (u_k, v_k)$ denote the motion vector found from QBMA. For each block the motion candidacy spread (MCS) $Spread_k$ is used as a weight in the first iteration of the regression:

$$\begin{bmatrix} w_1 & 0 & \cdots & 0 \\ 0 & w_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & w_K \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_K \end{bmatrix} = \begin{bmatrix} w_1 & 0 & \cdots & 0 \\ 0 & w_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & w_K \end{bmatrix} \begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ \vdots & \vdots & \vdots \\ x_K & y_K & 1 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_4 \end{bmatrix}$$

$$\begin{bmatrix} w_1 & 0 & \cdots & 0 \\ 0 & w_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & w_K \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_K \end{bmatrix} = \begin{bmatrix} w_1 & 0 & \cdots & 0 \\ 0 & w_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & w_K \end{bmatrix} \begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ \vdots & \vdots & \vdots \\ x_K & y_K & 1 \end{bmatrix} \begin{bmatrix} a_2 \\ a_3 \\ a_5 \end{bmatrix} \qquad \text{Eq 5-42}$$

$$w_k = \frac{1}{1 + \left(\frac{spread_k}{\sqrt{\text{var}(spread)}} \right)}$$

For each block, the reliability bears an inverse relation to the motion candidacy spread (MCS). The solution to the problem of scale [Pre-02] is provided by the spread measure. As the distribution of MCS is highly skewed, mean is not a good estimate; the median of the absolute deviation from the data median is employed instead. Hence w_k in each block k in Eq 5-42 is used as the scale factor. The set $\{a_{0,1}, a_{1,1}, a_{2,1}, a_{3,1}, a_{4,1}, a_{5,1}\}$ represents the initial estimate of the parameters. The solution to Eq 5-42 is similar to Eq 5-16. For each subsequent iteration n the global motion vector field $(u_{k,n}, v_{k,n})$ is estimated and a new weight based on the deviation of the estimation from the original motion vector field, $\mathbf{mvd}_{k,n}$, is used as the input to a robust estimate:

$$\begin{bmatrix} u_{1,n} \\ u_{2,n} \\ \vdots \\ u_{K,n} \end{bmatrix} = \begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ \vdots & \vdots & \vdots \\ x_K & y_K & 1 \end{bmatrix} \begin{bmatrix} a_{0,n} \\ a_{1,n} \\ a_{4,n} \end{bmatrix}; \quad \begin{bmatrix} v_{1,n} \\ v_{2,n} \\ \vdots \\ v_{K,n} \end{bmatrix} = \begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ \vdots & \vdots & \vdots \\ x_K & y_K & 1 \end{bmatrix} \begin{bmatrix} a_{2,n} \\ a_{3,n} \\ a_{5,n} \end{bmatrix} \qquad \text{Eq 5-43}$$

$$\mathbf{mvd}_{k,n} = \begin{bmatrix} u_{k,n} - u_k \\ v_{k,n} - v_k \end{bmatrix}$$

The iterative regression is then:

$$\begin{bmatrix} w_{1,n} & 0 & \cdots & 0 \\ 0 & w_{2,n} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & w_{K,n} \end{bmatrix} \begin{bmatrix} u_{1,n} \\ u_{2,n} \\ \vdots \\ u_{K,n} \end{bmatrix} = \begin{bmatrix} w_{1,n} & 0 & \cdots & 0 \\ 0 & w_{2,n} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & w_{K,n} \end{bmatrix} \begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ \vdots & \vdots & \vdots \\ x_K & y_K & 1 \end{bmatrix} \begin{bmatrix} a_{0,n} \\ a_{1,n} \\ a_{4,n} \end{bmatrix}$$

$$\begin{bmatrix} w_{1,n} & 0 & \cdots & 0 \\ 0 & w_{2,n} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & w_{K,n} \end{bmatrix} \begin{bmatrix} v_{1,n} \\ v_{2,n} \\ \vdots \\ v_{K,n} \end{bmatrix} = \begin{bmatrix} w_{1,n} & 0 & \cdots & 0 \\ 0 & w_{2,n} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & w_{K,n} \end{bmatrix} \begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ \vdots & \vdots & \vdots \\ x_K & y_K & 1 \end{bmatrix} \begin{bmatrix} a_{2,n} \\ a_{3,n} \\ a_{5,n} \end{bmatrix} \qquad \text{Eq 5-44}$$

$$w_{k,n} = \begin{cases} 1 & \left(\left\|\mathbf{mvd}_{k,n}\right\| < 1\right) \text{and} \left(\left\|\mathbf{v}_{k,n}\right\| < 1\right) \\ \left(1 - \left(\left\|\mathbf{mvd}_{k,n}\right\| / \left\|\mathbf{v}_{k,n}\right\|\right)^2\right)^2 & \left\|\mathbf{mvd}_{k,n}\right\| < \left\|\mathbf{v}_{k,n}\right\| \\ 0 & \text{otherwise} \end{cases}$$

The weight $w_{k,n}$ in Eq 5-44 uses robust statistics [Pre-02] to bring down the effects of outliers. Even though w_k in Eq 5-42 bears resemblance with $w_{k,n}$ the basic principles behind them are quite different. The weights w_k provide the confidence level of the local motion vector, which may or may not be due to global motion at all. The weights $w_{k,n}$, on the other hand, is similar to the Tukey's biweight M-estimator scaled by the block's observed motion vector. The additional term (top-most condition) is used to remove the influence of those blocks without significant motion.

$$\left\|\mathbf{mvd}_{k,n}\right\| = \left\langle \mathbf{mvd}_{k,n}, \mathbf{mvd}_{k,n} \right\rangle = \left(u_{k,n} - u_k\right)^2 + \left(v_{k,n} - v_k\right)^2 \qquad \text{Eq 5-45}$$
$$\left\|\mathbf{v}_{k,n}\right\| = u_{k,n}{}^2 + v_{k,n}{}^2$$

Another improvement based on SAD-map is the use of motion candidacy points. After each step of the iteration, a global motion field is generated. The standard procedure is then to compare this estimated field with the observation field obtained from the initial BMA or QBMA process. In the proposed method, the observation field is first adapted prior to this comparison. This is made possible by the motion candidacy set, which contains a set of motion vectors for each block. At each iteration n, the observed motion vector field $\{\mathbf{v}_k : k = 1 \ldots K\}$ is replaced by an adapted motion vector field $\{\bar{\mathbf{v}}_{k,n} \in Cand(k) : k = 1 \ldots K\}$ whose members are selected from the motion candidacy set of vectors which is closest to the estimated field:

$$\bar{\mathbf{v}}_{k,n} = \arg\min_{\mathbf{v} \in Cand(k)} \left\|\mathbf{v} - \mathbf{v}_{k,n}\right\| \qquad \text{Eq 5-46}$$

The new observation field is then used to replace the original observation field in Eq 5-44. Figure 5.6 shows the adaptation of the new observed motion vector in a block with multiple candidates. By shifting to a new observed motion vector which is the closest to the estimated vector, blocks with a large candidate set due to aperture problem can provide enough support to many parameter sets as long as these sets induce a motion similar to one of the members within the candidate set. The effect of this adaptation is similar to statistical relaxation methods which tend to bring the algorithm out of local minimum.

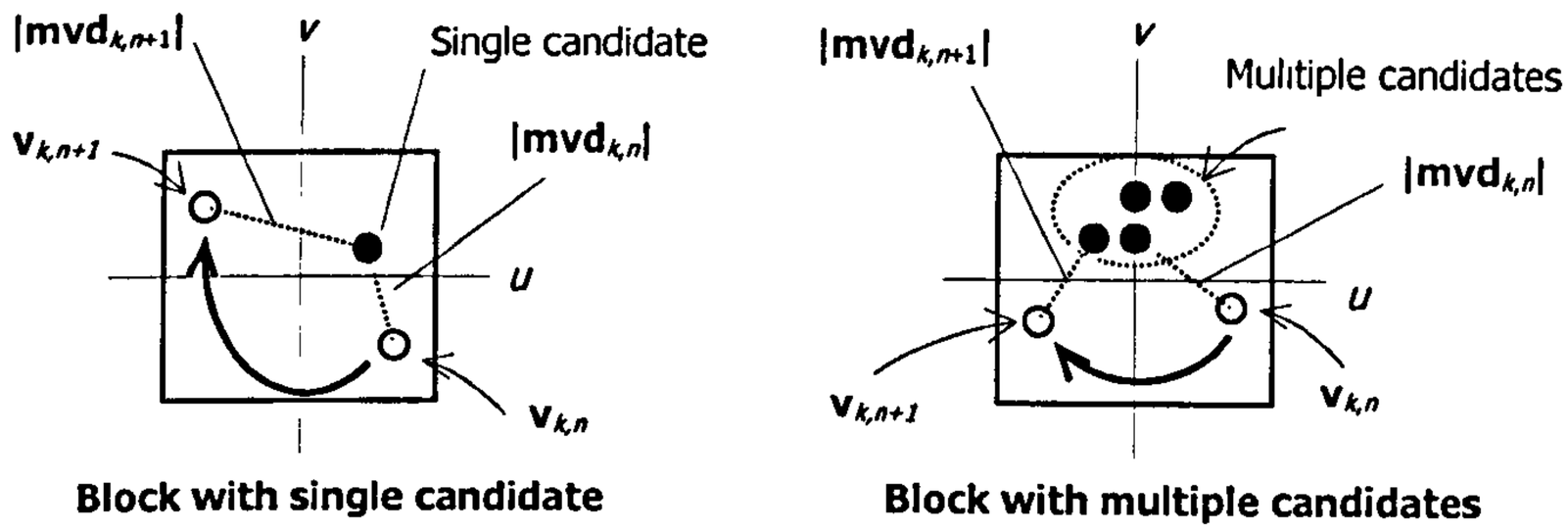


Figure 5.6. Illustration of observation field adaptation. The motion vector in the right block is changed to the member within the candidacy set closest to the estimated vector.

Termination of the iteration process is triggered when the parameters changes within the threshold values:

$$\begin{aligned} &\left|a_{0,n+1} - a_{o,n}\right| < a_{0,threshold} \\ &\left|a_{1,n+1} - a_{1,n}\right| < a_{1,threshold} \\ &\vdots \\ &\left|a_{5,n+1} - a_{5,n}\right| < a_{5,threshold} \end{aligned} \qquad \text{Eq 5-47}$$

We set the translational threshold values as $a_{5,threshold} = a_{4,threshold} = (64.0)^{-1}$. The other 4 thresholds values are related to scaling and rotation, ($a_{0,threshold} = a_{1,threshold} = a_{2,threshold} = a_{3,threshold}$). They are selected so that they produce the translational threshold at the border of the picture. For QCIF sequences (176×144), the threshold value is set as $(64.0)^{-1}/128 = (8192.0)^{-1}$, whereas for CIF sequences (352×288), the value of $(64.0)^{-1}/256 = (16384.0)^{-1}$ is used. In the rare case of a divergence, a maximum iteration of 16 is imposed on the algorithm.

5.5 Simulation Results

5.5.1 Choice of Global Motion Model

As the number of parameters increases, the GME should better model global motion provided the dominant motion is not too severely masked by local motion. However, this accuracy has to be weighed against the complexity required to compute the model parameters. Out of the eight test sequences used in the previous chapters, six of them contains a global motion and some foreground moving object. These six sequences of both QCIF and CIF sizes (at 10 fps and 30 fps respectively) are simulated using the traditional regression-based global motion estimation with the seven motion models and the control (simple BMA):

- NONE = local motion estimation by BMA.
- T = translations;
- TZ = zoom + translation;
- TZR = quasi affine (zoom+rotation+translation);
- AFF = affine;
- PERS = perspective;
- H263 = bilinear (as described in the Reference Picture Resample option of H.263+);
- PARA = parabolic.

The effectiveness of a motion model is measured by the amount of motion vector field entropy reduced by the elimination of the global motion components. Hence the entropy of the residual motion field is used as an indicator of global model effectiveness, as shown in Figure 5.7. Figure 5.8 provides a comparison of the processing time required for various models. Figure 5.9 and Figure 5.10 depicts the same graph of the QCIF sequences. For the entropy plots, all models are shown to out-perform the BMA to different degrees. The trend of each sequence is similar in both CIF and QCIF versions. The BUS, MOBILE and TABLE sequences are best modelled by the 3-parameter TZ model as the sequence has a prominent zoom and pan component. The COAST sequence is a pure panning sequence; as a result, the pure T model is sufficient. In fact, more complex models are less effective due to the amount of noise contained in the motion vector field (attributable to the water reflection). The FOREMAN sequence performs equally well with all the models, except the PERS model, which due to the non-linear nature is where a_6 and a_7 are very susceptible to noise. Surprisingly, the T model seems to outperform the other models slightly. This is probably due to a panning motion towards the end of the sequence. AS for the STEFAN sequence, the three models (T, TZ and AFF) outperform the other models.

The TZR model does not seem to perform as well as the T and TZR models. This may be due to the fact that all sequences tested have no rotational component (a_1 of the TZR model). Even if the

rotational component does exist, the model will still not be able to provide an accurate estimate of the field. This is due to the fact the most sequences do no have exact 1:1 aspect ratio. Unlike the zoom factor which is invariant to the aspect ratio, the rotation component is not. A better model would be the affine model (AFF). Furthermore, the x- and y- component of the TZR is inseparable while the AFF model is. This makes AFF model slightly computationally less intensive then the TZR model. Hence AFF should be preferred over the TZR model.

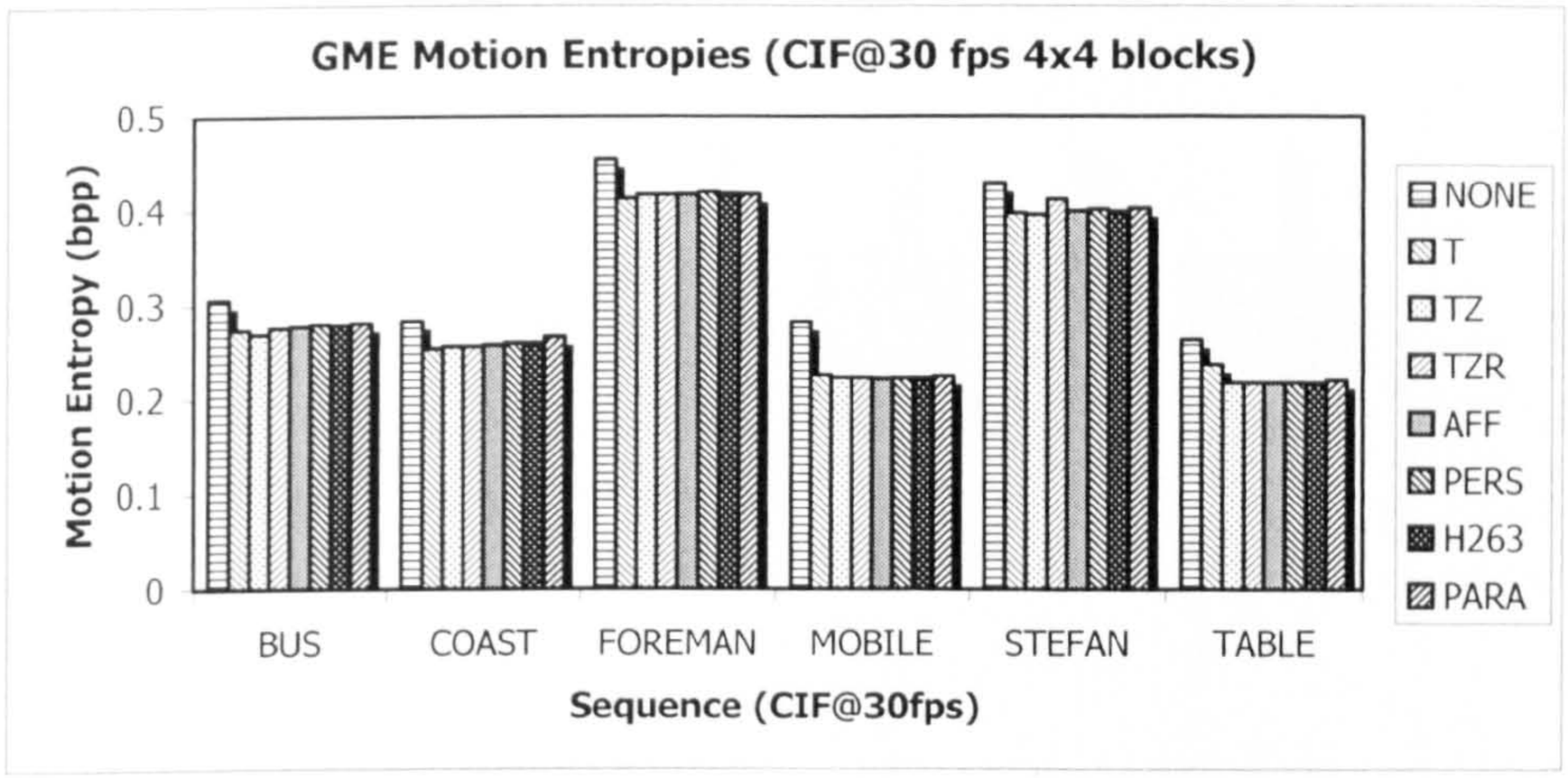
From the processing time plots, all models requires about same processing time (1.5 seconds for CIF and 0.4 seconds for QCIF sequences) except the perspective (PERS) model, which takes a substantially long time to process. This is mainly due to the requirement of the floating point division in the model. Even the PARA model which has 12 parameters takes much less time. In any case, block-based GME does not benefit much by using global motion models more complex then the affine (AFF) model. This premise only holds for applications relevant to this chapter; more accurate models may benefit other applications like pixel-based motion segmentation.

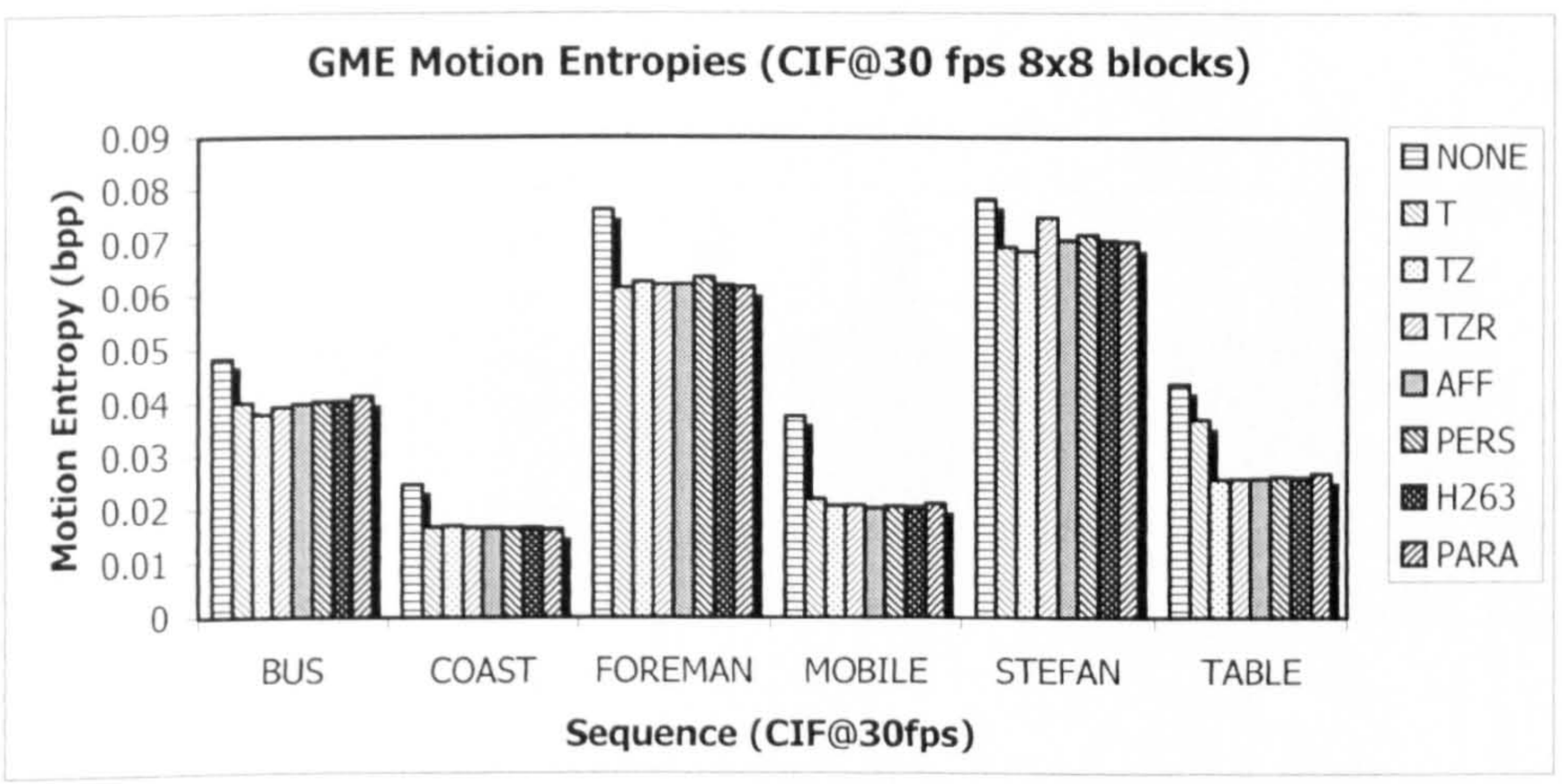
Amongst the three models with the largest number of parameters (PERS, H263 and PAR), the H263 model performs best. The bilinear nature of H263 also makes this model easy to implement and a fixed-point version is actually recommended in the H.263+ standard. An added advantage is that less complex model (T, TZ, TZR and AFF) are special cases of the H263 model. Hence the H263 model can provide a common model representation for coding purposes.

In conclusion, this thesis recommends the use of the following models in increasing complexities:

- TZ model – for entry level applications where processing power is limited and timing is crucial.
- AFF model – for applications which requires intermediate complexity. This is also the basic model to use for scenes with rotational motion, and for motion segmentation applications.
- H263 model – for more applications with higher motion model complexity.

Finally, all three models can be represented with the H263 model where some of the parameters are set to zero.





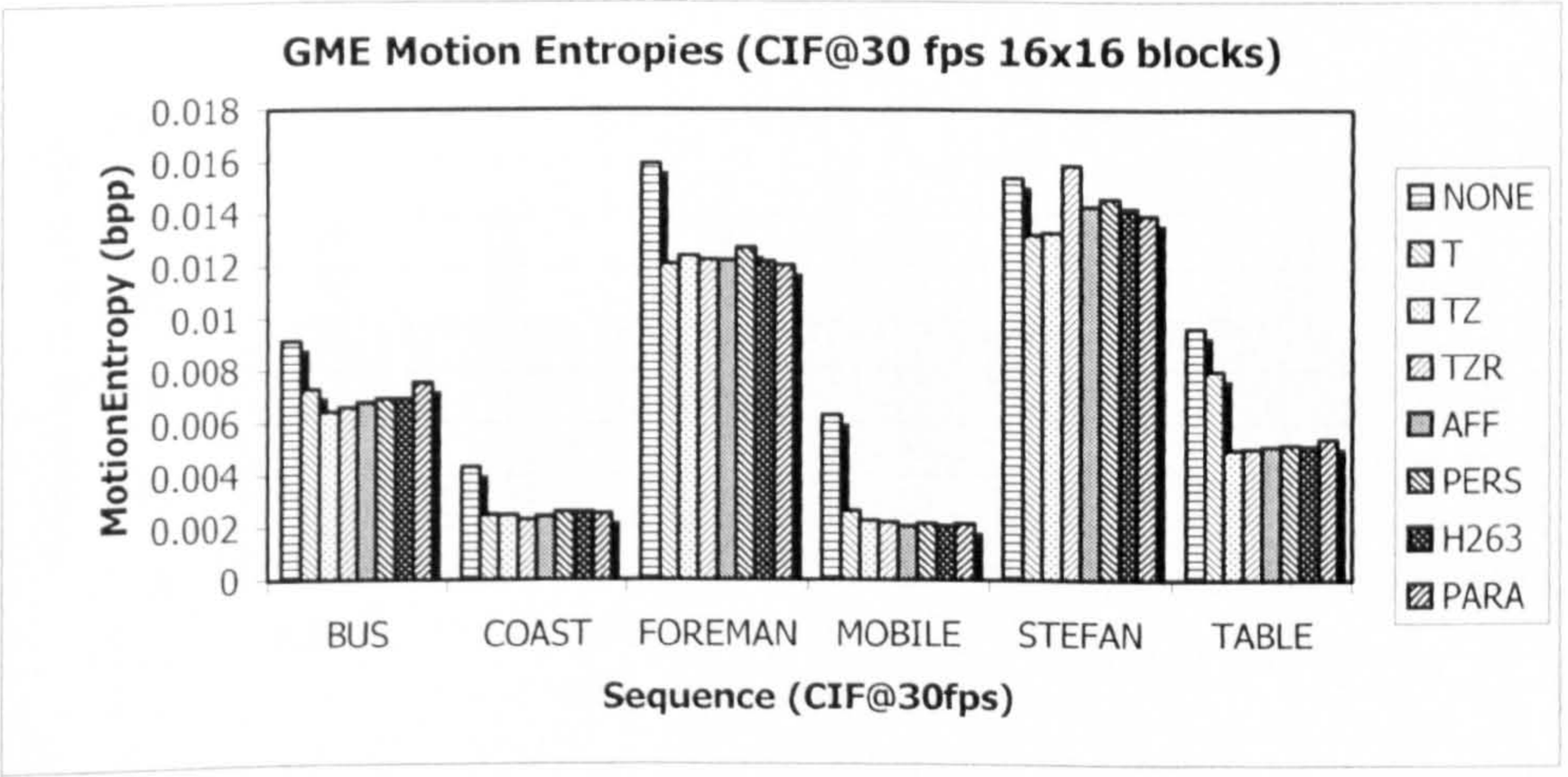


Figure 5.7. Prediction performance of various GME models on 6 CIF@30fps sequences with various block sizes: top: 4×4; centre: 8×8; bottom: 16×16

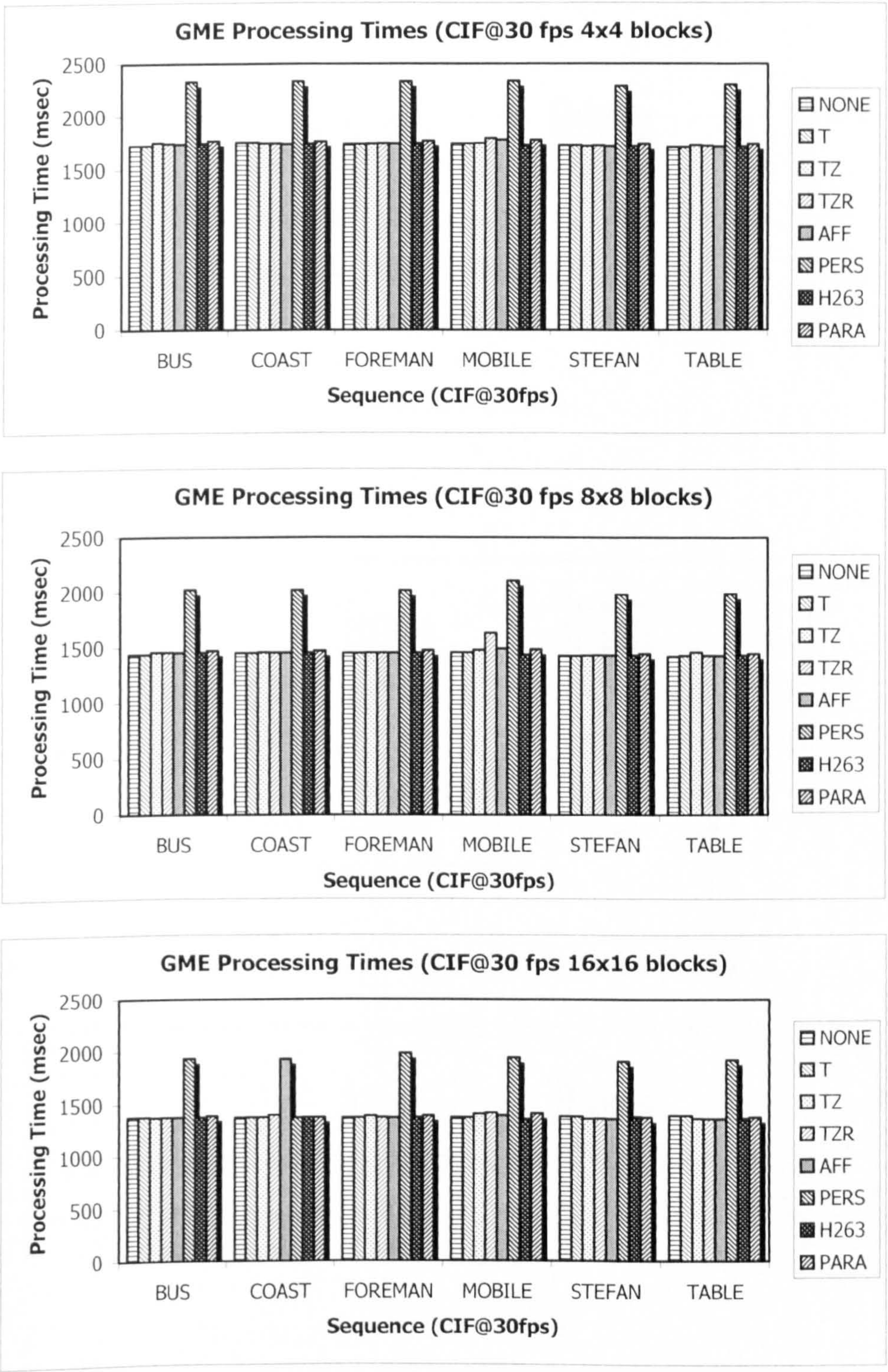


Figure 5.8. Processing times of various GME models on 6 CIF@30fps sequences with various block sizes: top: 4×4; centre: 8×8; bottom: 16×16

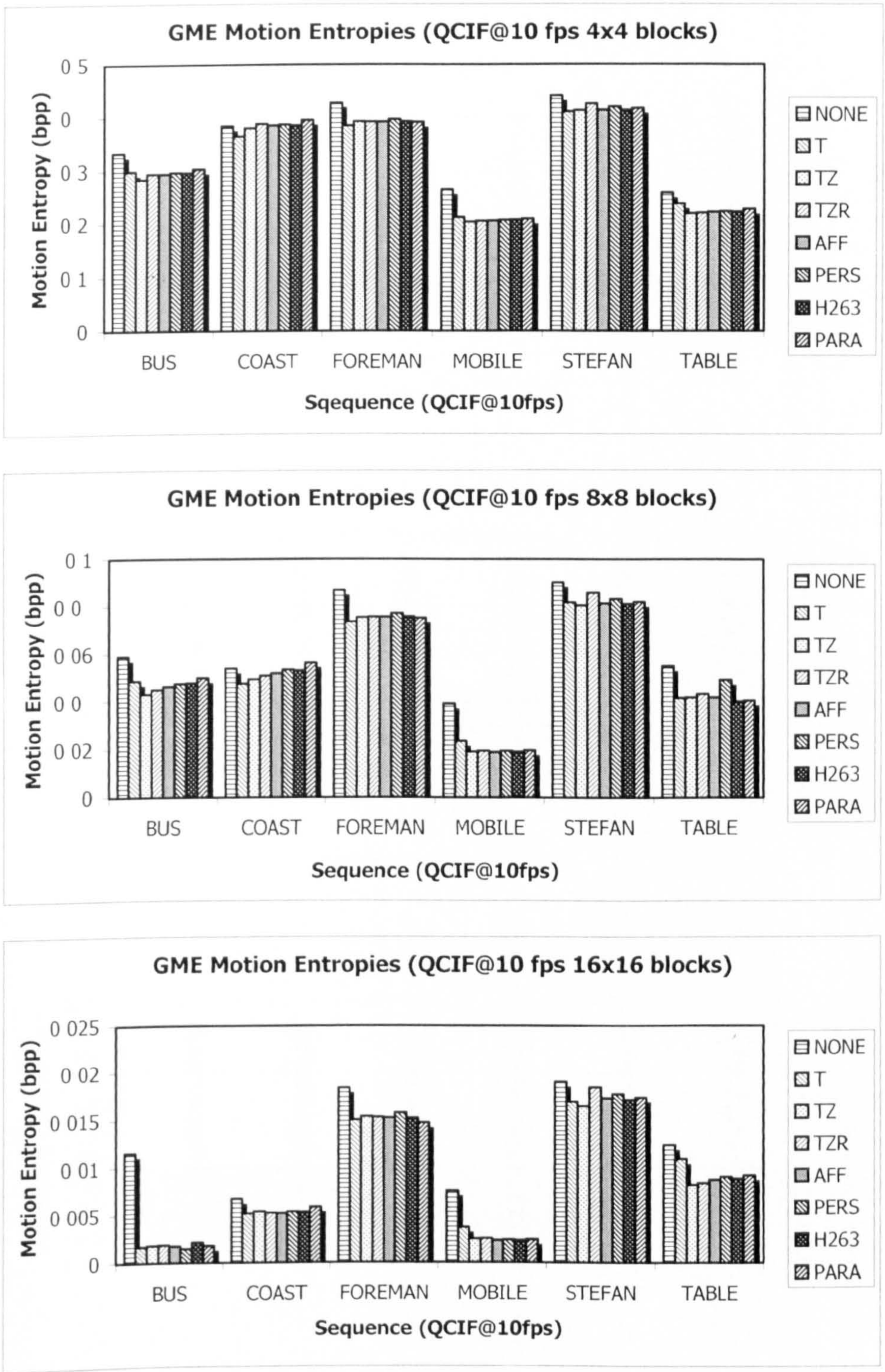


Figure 5.9. Prediction performance of various GME models on 6 QCIF@10fps sequences with various block sizes: top: 4×4; centre: 8×8; bottom: 16×16

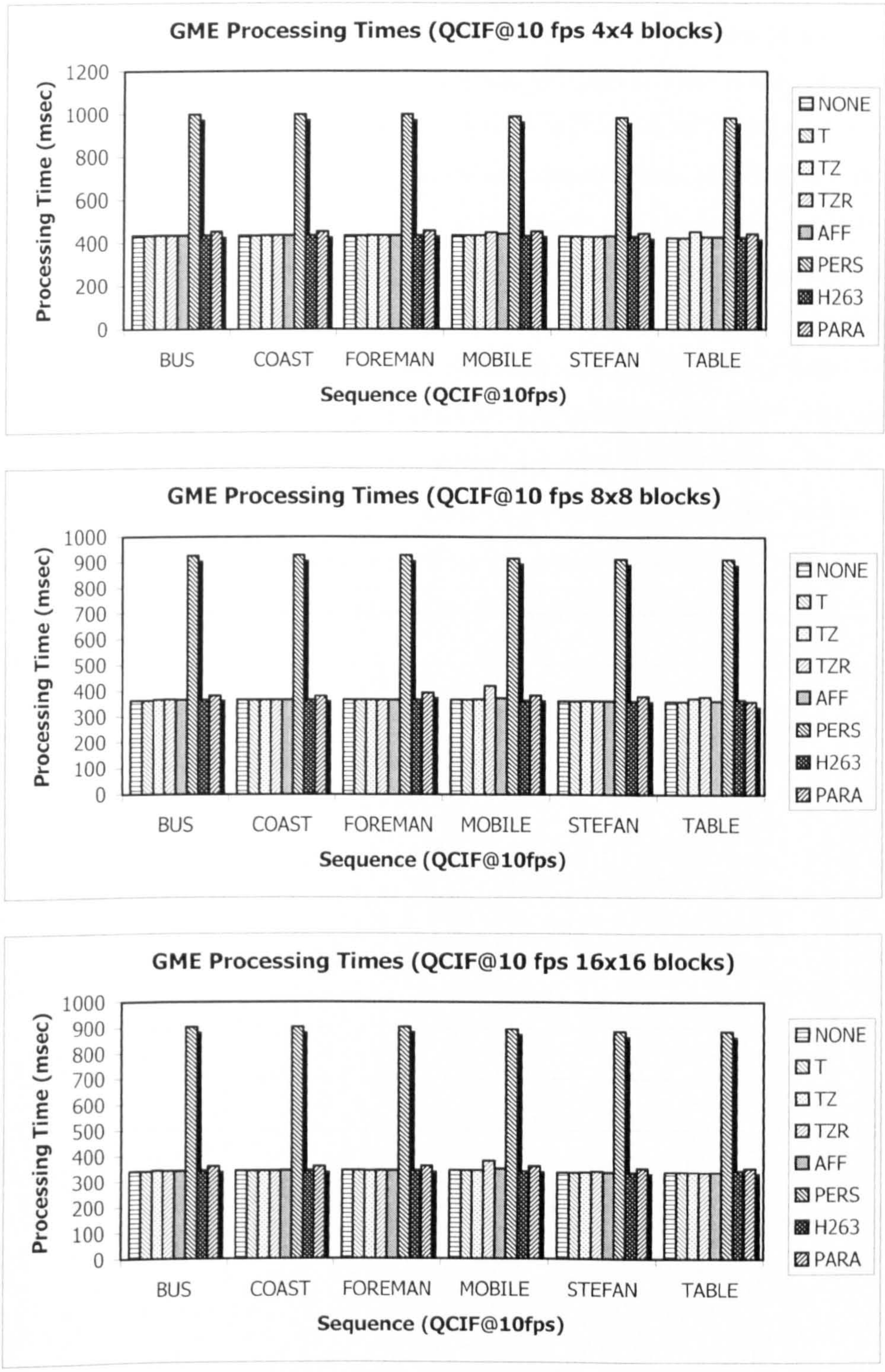


Figure 5.10. Processing times of various GME models on 6 QCIF@10fps sequences with various block sizes: top: 4×4; centre: 8×8; bottom: 16×16

5.5.2 Effect of Block Size for BMA used in GME

One of the considerations in BMA is the block size. Smaller block size provides better matched locally, but its motion vector field suffers from the general aperture problem as there is less texture to match per block. Large block sizes are more robust to noise, but have higher risk of having multiple objects in a single block. Hence compromise has to be made when selecting the block size in BMA. This is more so when the vector field is to be used for GME, as a wrong block size can make the whole GME result useless. Block sizes of 4, 8 and 16 are tested CIF and QCIF test sequences to determine which block size is optimal. The combined entropies of the motion-compensated residues and the residual motion vectors are used as an indicator of compression efficiency. Figure 5.11 and Figure 5.12 show the combined entropies of the six sequences (QCIF and CIF respectively) using GME with different block sizes. 3 models as recommended in the previous section are used. It is clear from the figures that for QCIF sequences, block-based GME on QCIF sequences are best done with 4×4 blocks whereas CIF sequences benefit most with 8×8 blocks, regardless of which model being used. Henceforth, for subsequence simulations with QCIF sequences, BMA based on 4×4 blocks will be used; CIF sequences will be based on 8×8 blocks.

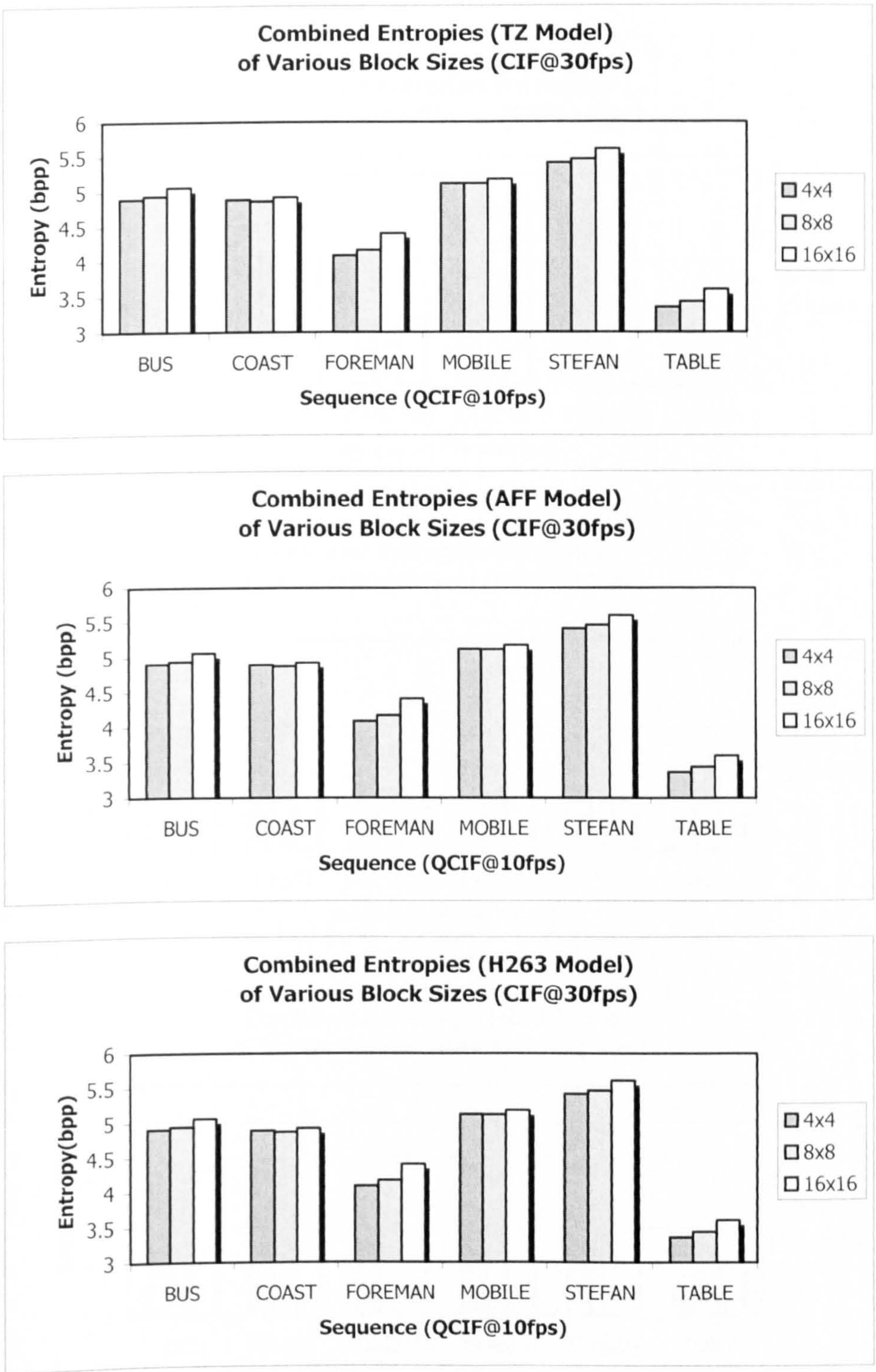
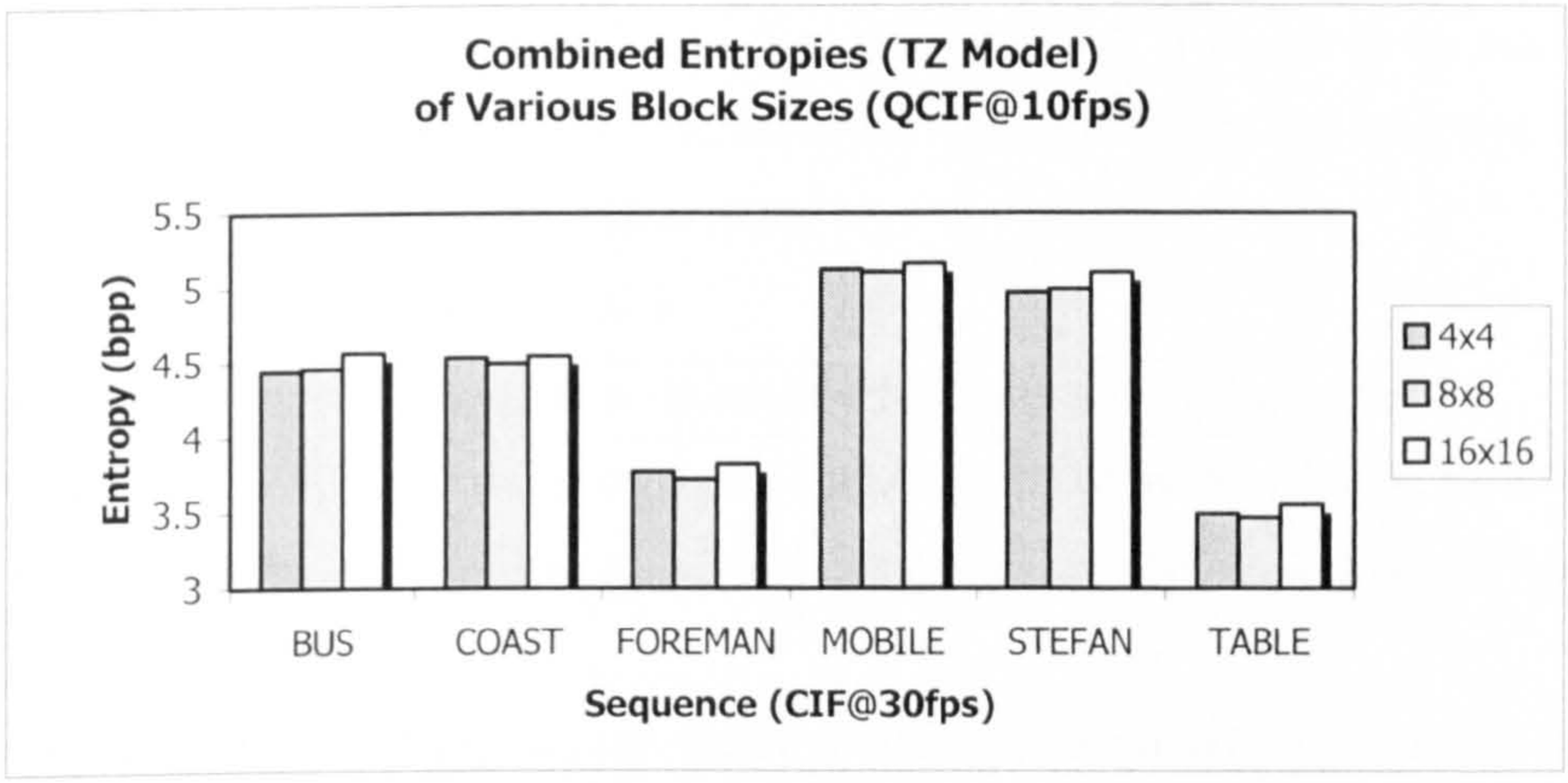
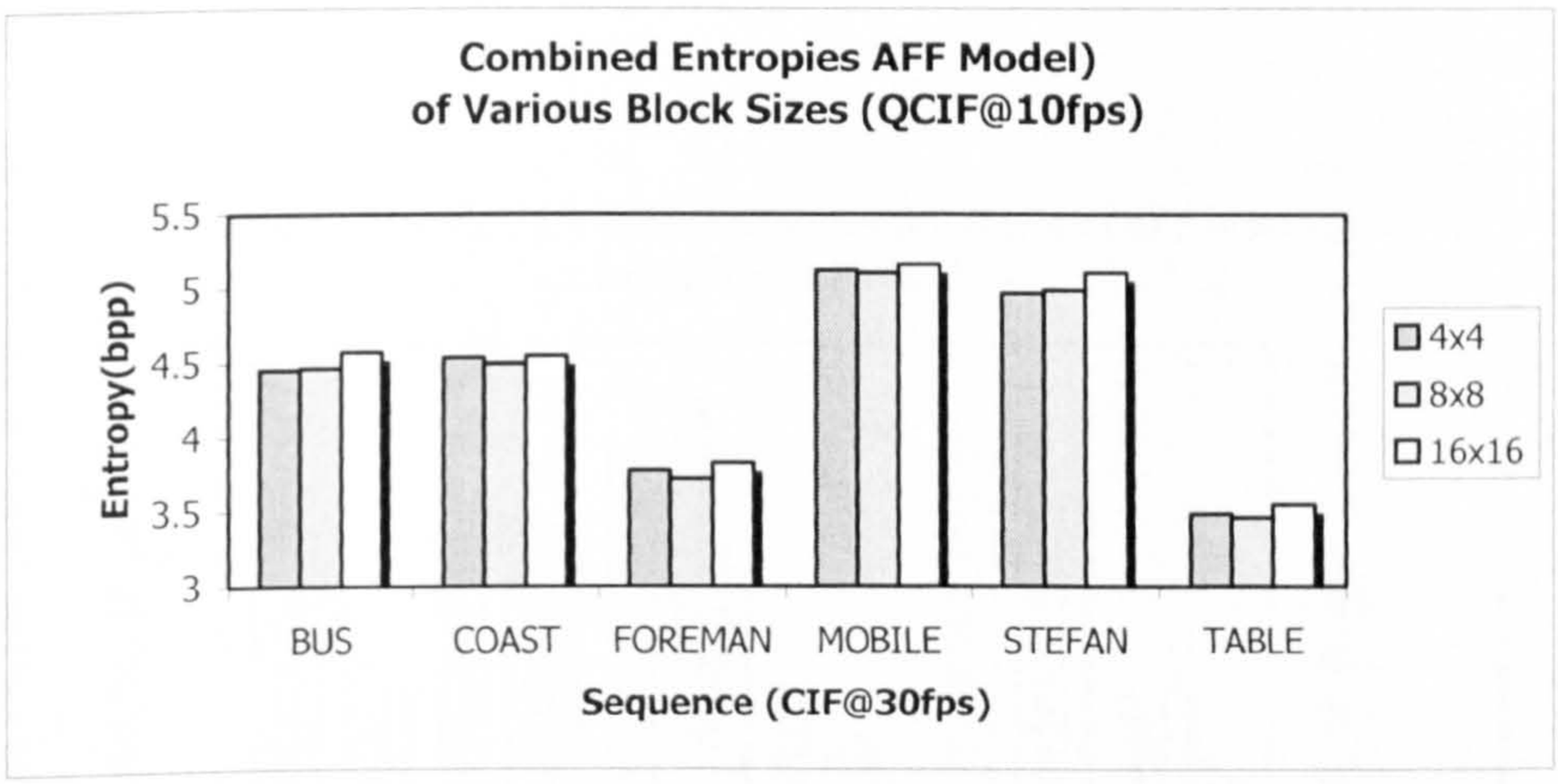


Figure 5.11. Combined entropies of 6 QCIF@10fps sequences with various block sizes using: TZ model (top), AFF model (centre) and H263 (bottom).





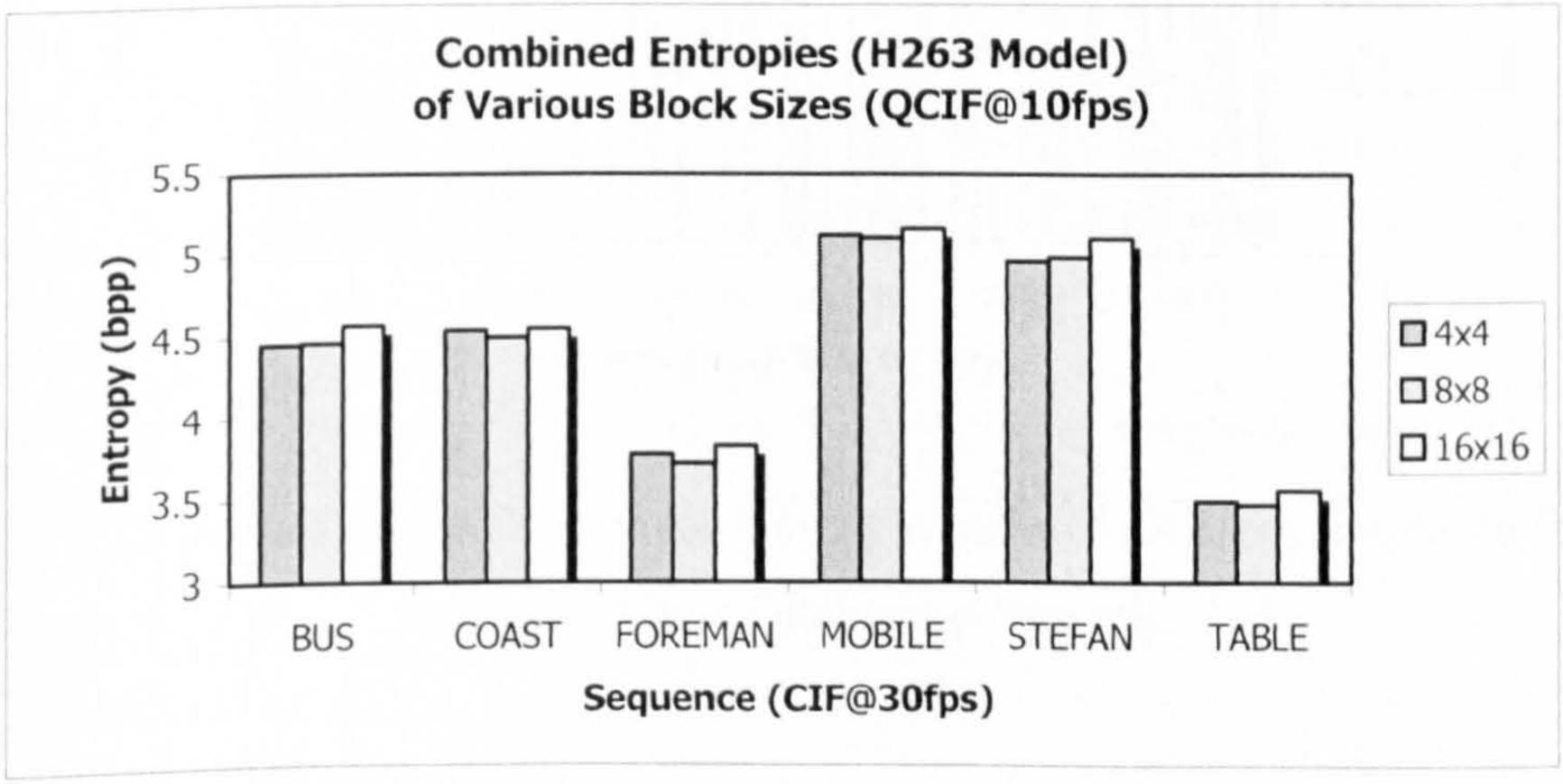


Figure 5.12. Combined entropies of 6 CIF@30fps sequences with various block sizes using: TZ model (top), AFF model (centre) and H263 (bottom).

5.5.3 Various Reliability Measures in Regression-based GME (RGME)

With regression-based GME (RGME), this thesis recommended the use of MCS as the reliability measure to improve the single-step regression, as depicted in Eq 5-42. To find out the merits of various reliability measures described in Chapter 4, simulations are conducted using the following measures:

- BMA – benchmark where no GME is used.
- NONE – no reliability measure used.
- GRAD – reliability measure based on image intensity gradients, Eq 4-9(a)
- SADMIN – reliability measure based on minimum SAD of the each block, Eq 4-9(b)
- MVS – reliability measure based on local smoothness of the motion vector field, Eq 4-9(c)
- MCS – proposed reliability measure based on the motion candidacy spread (MCS), Eq 5-42.

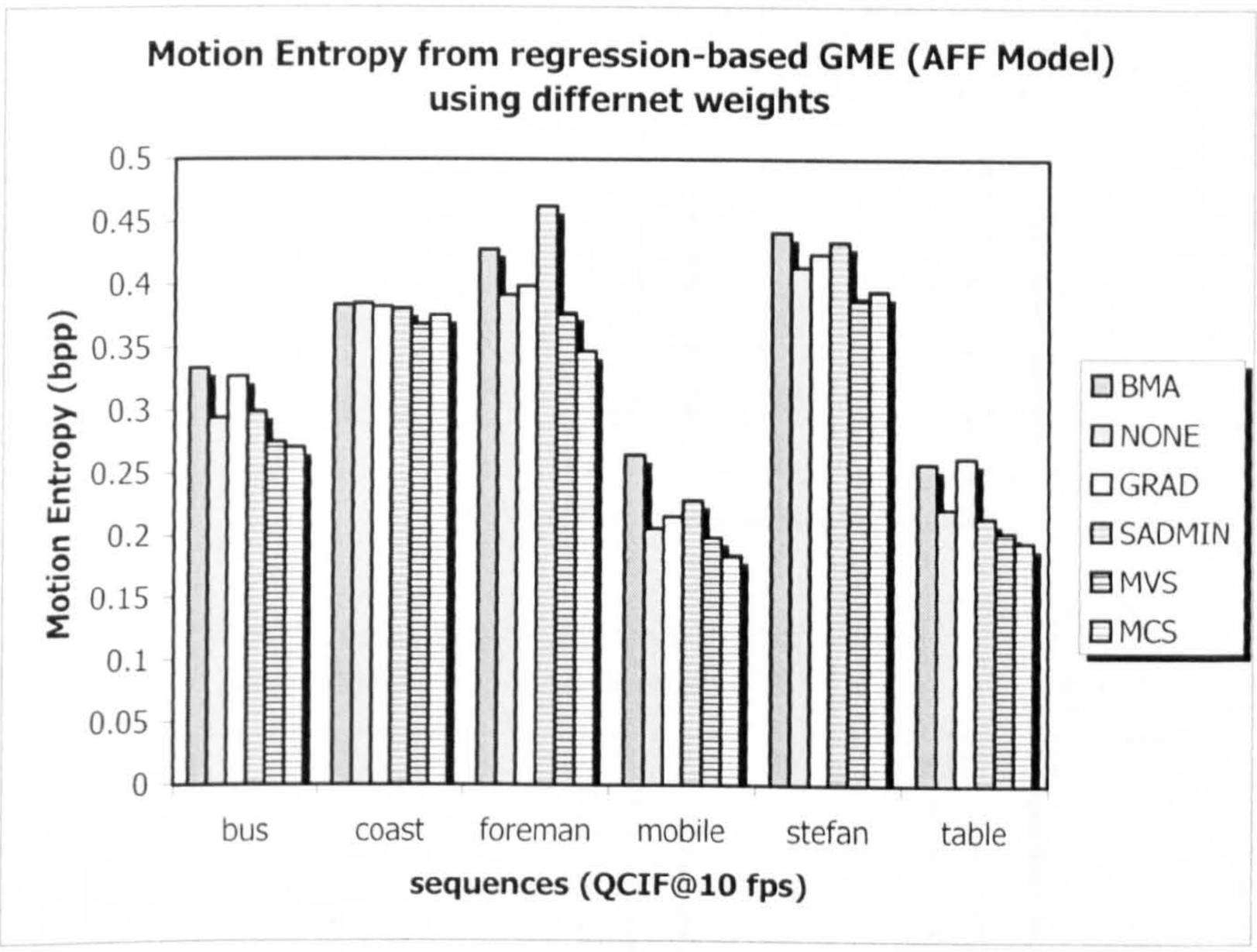


Figure 5.13. A chart showing the motion entropy resulting from regression-based GME using different reliabilities as weights.

Simulation results on the six QCIF sequence using the affine model are shown in Figure 5.13. The texture-based and minimum SAD-based measures are poor measures; they sometimes produce worst results than the non-weighted algorithm does. Reliability measure based on local motion smoothness produces good results; the proposed MCS-based measure outperforms all other measures in all the sequences, except for two sequences, COAST.QCIF and STEFAN.QCIF. The COAST.QCIF sequence

produces less smooth sequence due to the speckle reflection of the water surface, which creates a lot of noise in the motion field, the MCS weights tends to 'over-smooth' the field, thus producing more motion residues. The STEFAN.QCIF sequence, on the other hand, has many fast moving components and motion estimation fails at such places at 10 fps, and MCS is not able to create a better estimate of the global motion. As a whole, the MCS weight is a superior weight for regression-based GME compared with the other weights used in the simulations. Subsequent simulations will be based on this weight measure.

5.5.4 SAD-map Iterative Regression-based GME (SIRGME)

Using the result of the RGME above, iteration was carried out as described in Eq 5-44. The main concern is the complexity involved in the iterations and the number of iterations required to reach a desired tolerance.

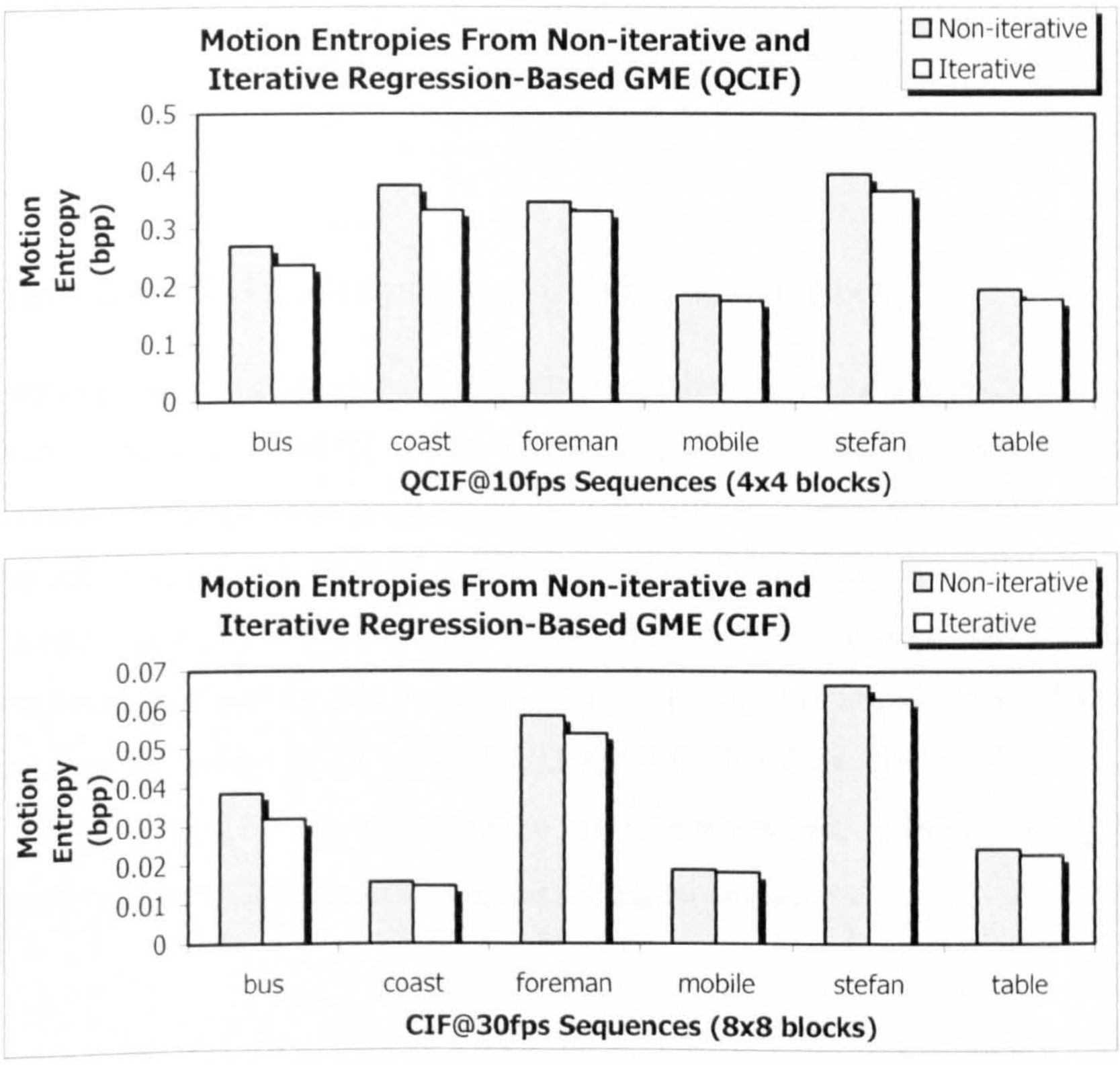


Figure 5.14. A chart showing the motion entropy resulting from non-iterative and iterative regression-based GME. Affine model is used. Top: QCIF sequences; bottom: CIF sequences.

As is evident from Figure 5.14, iterative regression reduces motion entropies of all test sequences. On average, a 10% performance can be expected for both QCIF and CIF sequences. The average number

of iterations and iteration times for the sequences are summarised in Table 5.4. The numbers of iterations for all sequences are less than 11, and with about 1 millisecond processing time per iteration step, it is highly applicable in real-time scenarios.

Table 5.4 Average iterations and processing time for iterative-regression-based GME using sparse motion vector field with QBMA.

	QCIF@10fps		CIF@30fps	
	Average Number of Iterations	**Time taken for iterations (msec)**	**Average Number of Iterations**	**Time taken for iterations (msec)**
Bus	6.35714	6.17347	4.5	4.7651
Coast	7.27551	6.82653	3.50671	3.83221
Foreman	5.68367	5.30612	4.70134	4.90604
Mobile	4.09184	4.53061	3.59396	4.05705
Stefan	10.1939	9.2551	10.9161	10.3523
Table	3.73469	4.32653	3.20134	3.72148

5.5.5 Variation of GME Parameters in Test Sequences

Lastly, this chapter presents the variation of the motion parameters throughout each test sequence. Only the QCIF version is presented and the affine model parameters are shown. As recommended earlier, this model is a compromise between complexity and accuracy of various models. Figure 5.15 to Figure 5.20 shows the affine parameters of the six test sequences. The left charts records the four scale and rotation parameters $\{a_0, a_1, a_2, a_3\}$ whereas the right charts show the translational parameters $\{a_4, a_5\}$. For all the sequences, the scaling and rotation (along with the skew and stretch factors) lie within ± 0.08; in most cases, the values lie far below ± 0.02. On the other hand, the translational parameters lie within ± 12.0. These limits serve as a guide-line to estimate the boundary and dynamic range of use in the Hough Transform-based GME described in the following chapter.

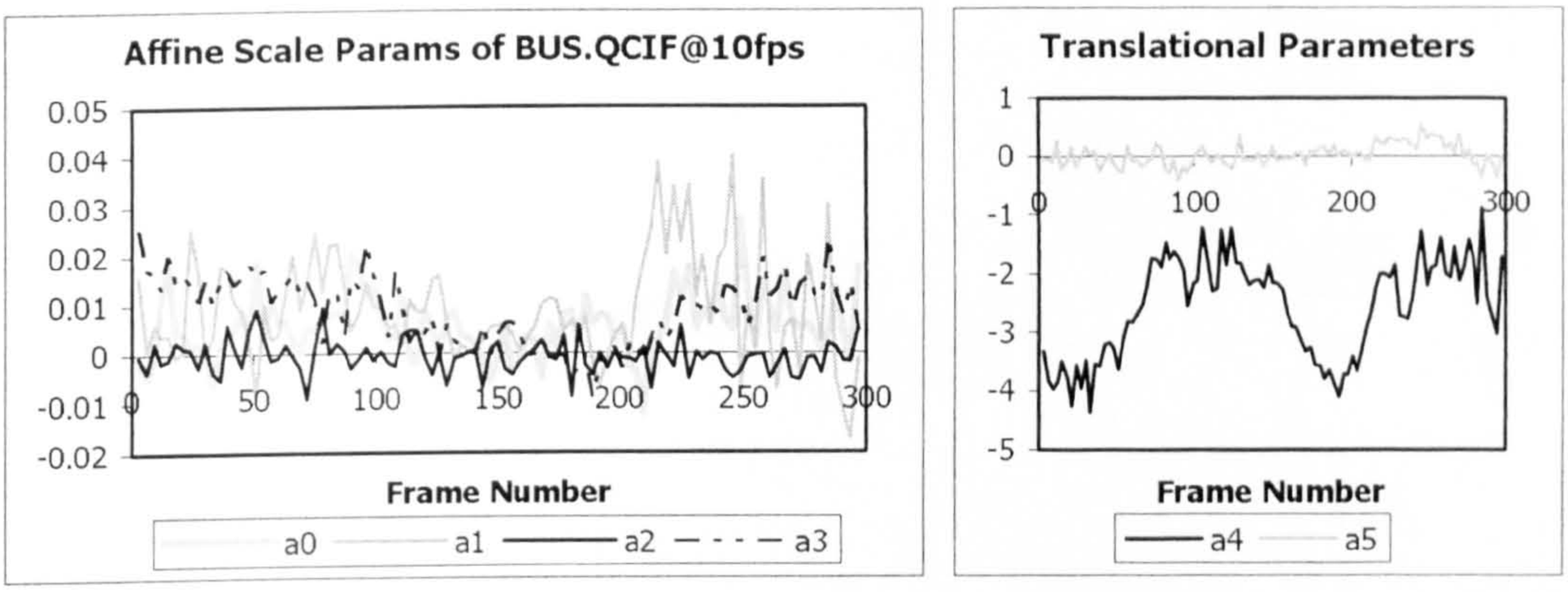


Figure 5.15.Affine global motion parameters of BUS.QCIF

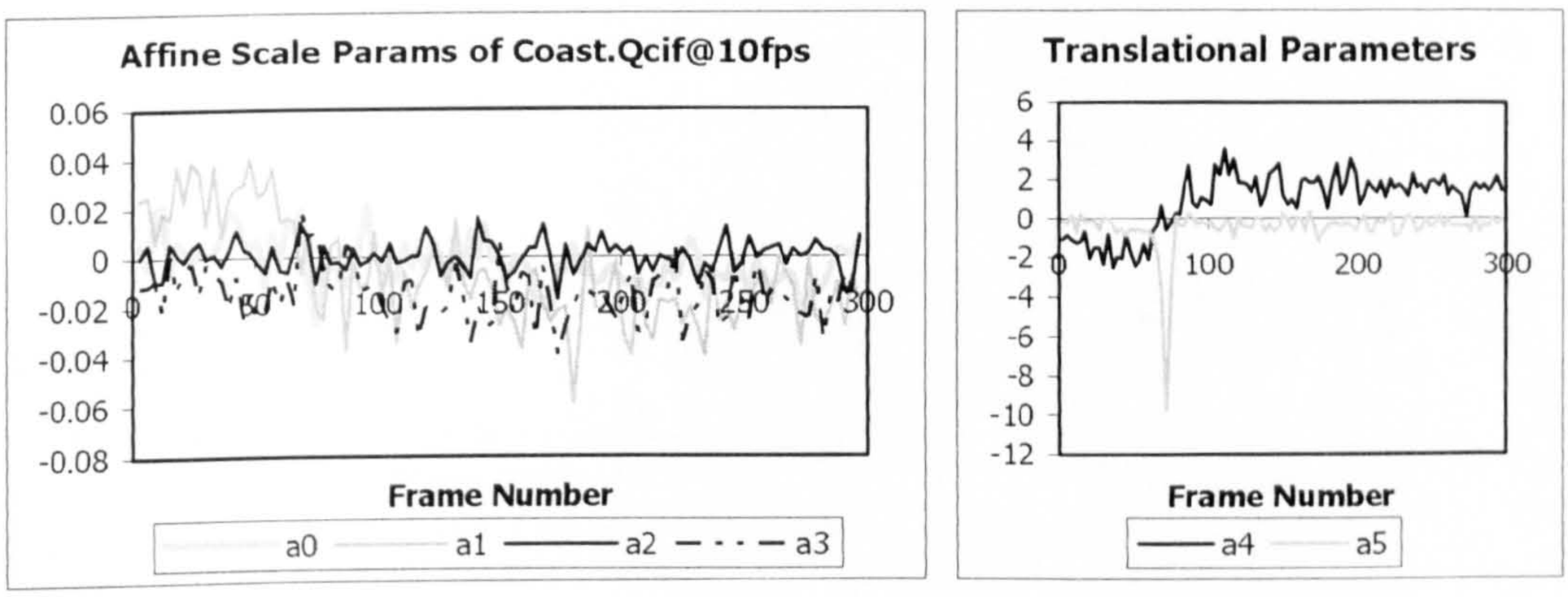


Figure 5.16.Affine global motion parameters of COAST.QCIF

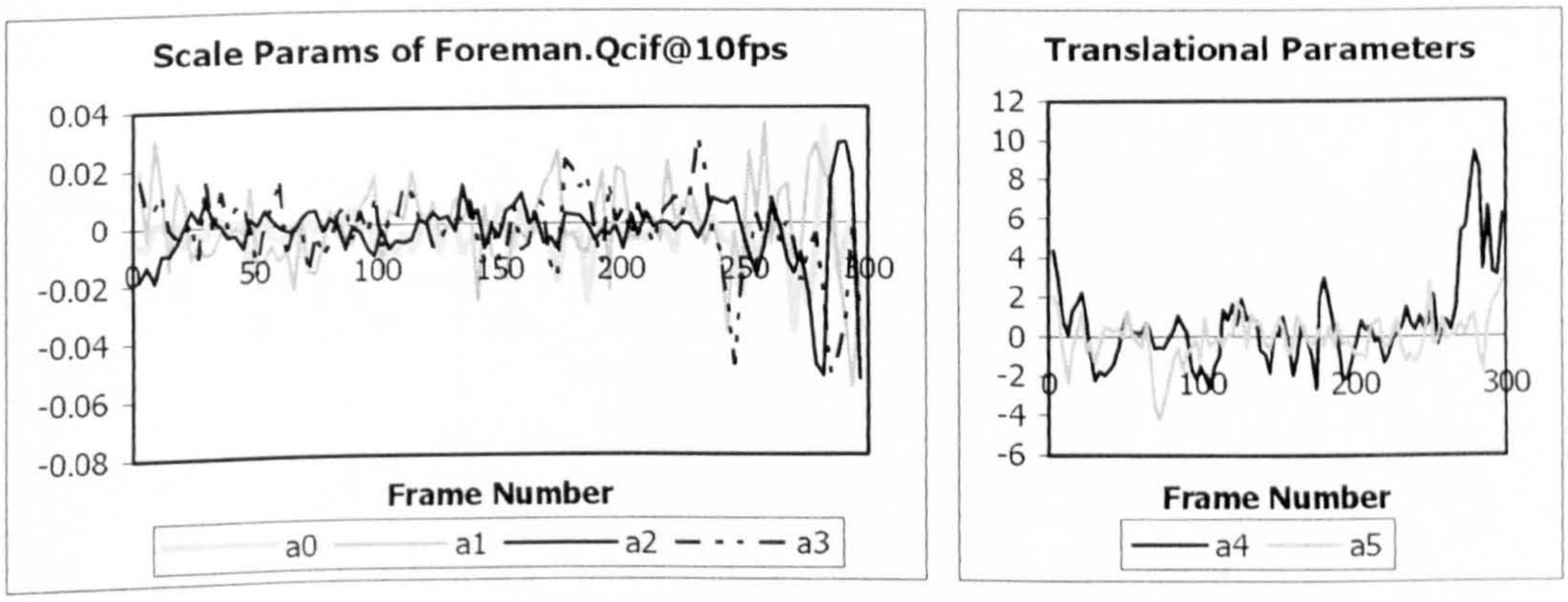


Figure 5.17.Affine global motion parameters of FOREMAN.QCIF

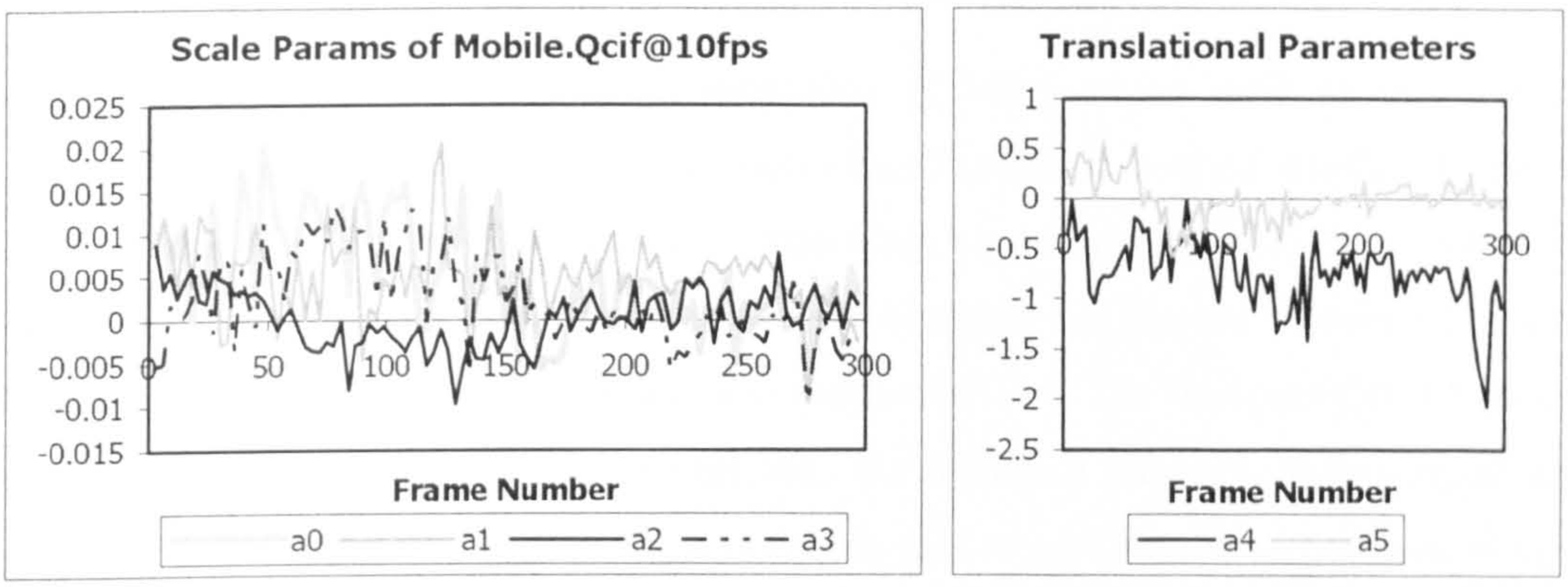


Figure 5.18.Affine global motion parameters of MOBILE.QCIF

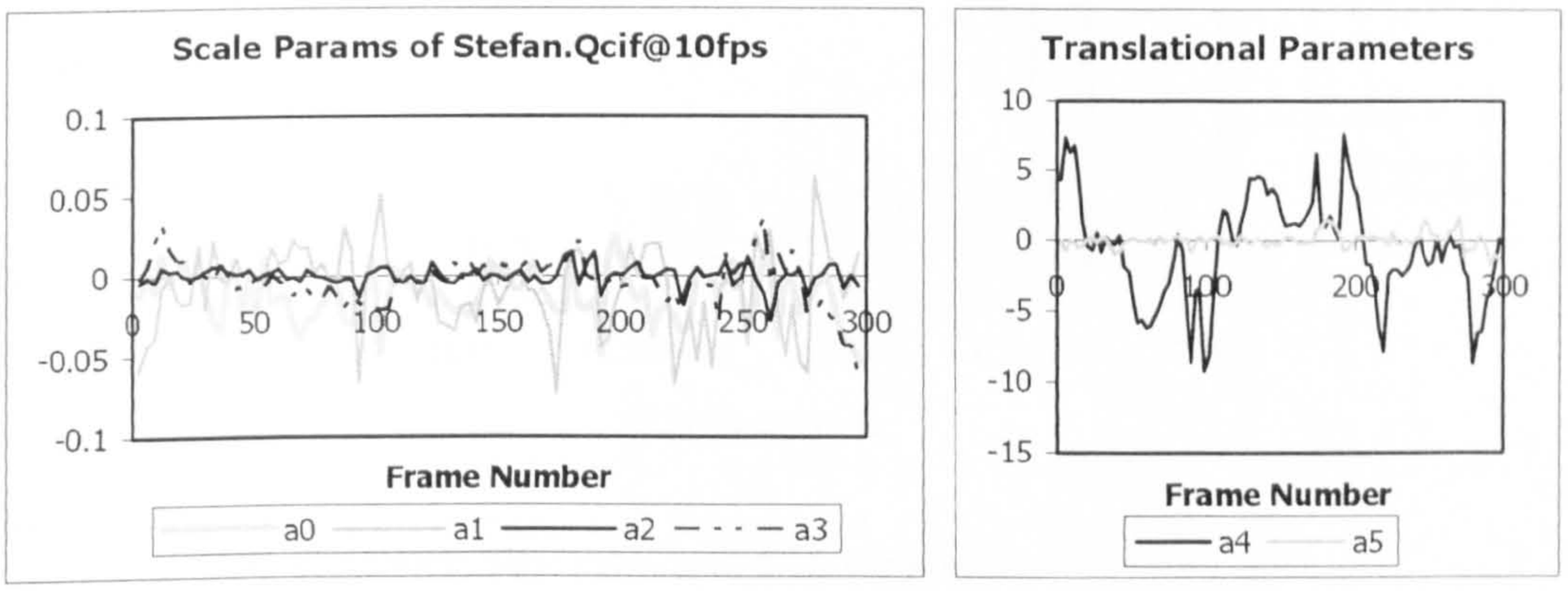


Figure 5.19.Affine global motion parameters of STEFAN.QCIF

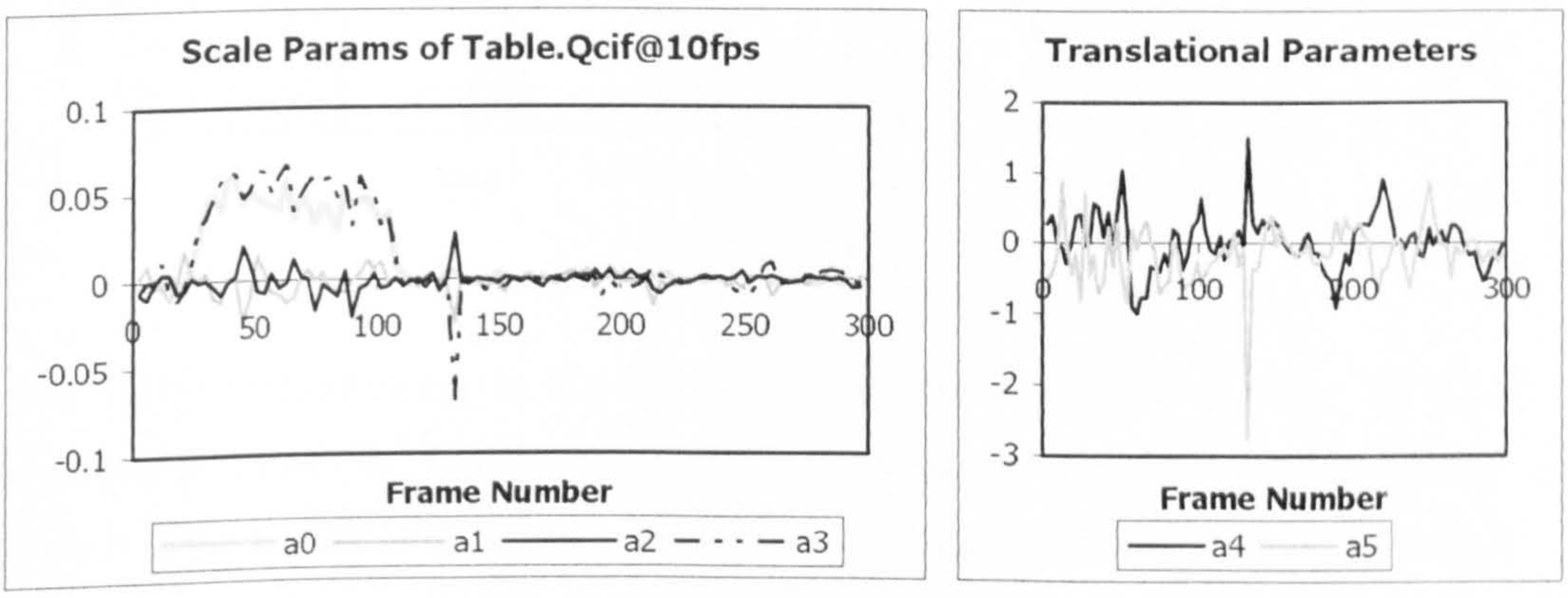


Figure 5.20.Affine global motion parameters of TABLE.QCIF

5.5.6 Performances of GME-based Displaced Inter-frame Prediction

This section investigates how global motion estimation (GME) can be used to improve coding efficiencies. Collectively, the coding based on motion estimation is termed displaced inter-frame coding. The first scheme is the Queue-based Block Matching Algorithm (QBMA). The next scheme is to represent the motion vector field with a global motion parameter set plus the remaining vector field using SIRGME. The last scheme performs a two-pass motion QBMA. The first pass QBMA produces a motion vector field which is used to perform SIRGME; the reference picture is then warped with the resulting parameters. The warped reference frame is used in turn to perform the second pass of QBMA.

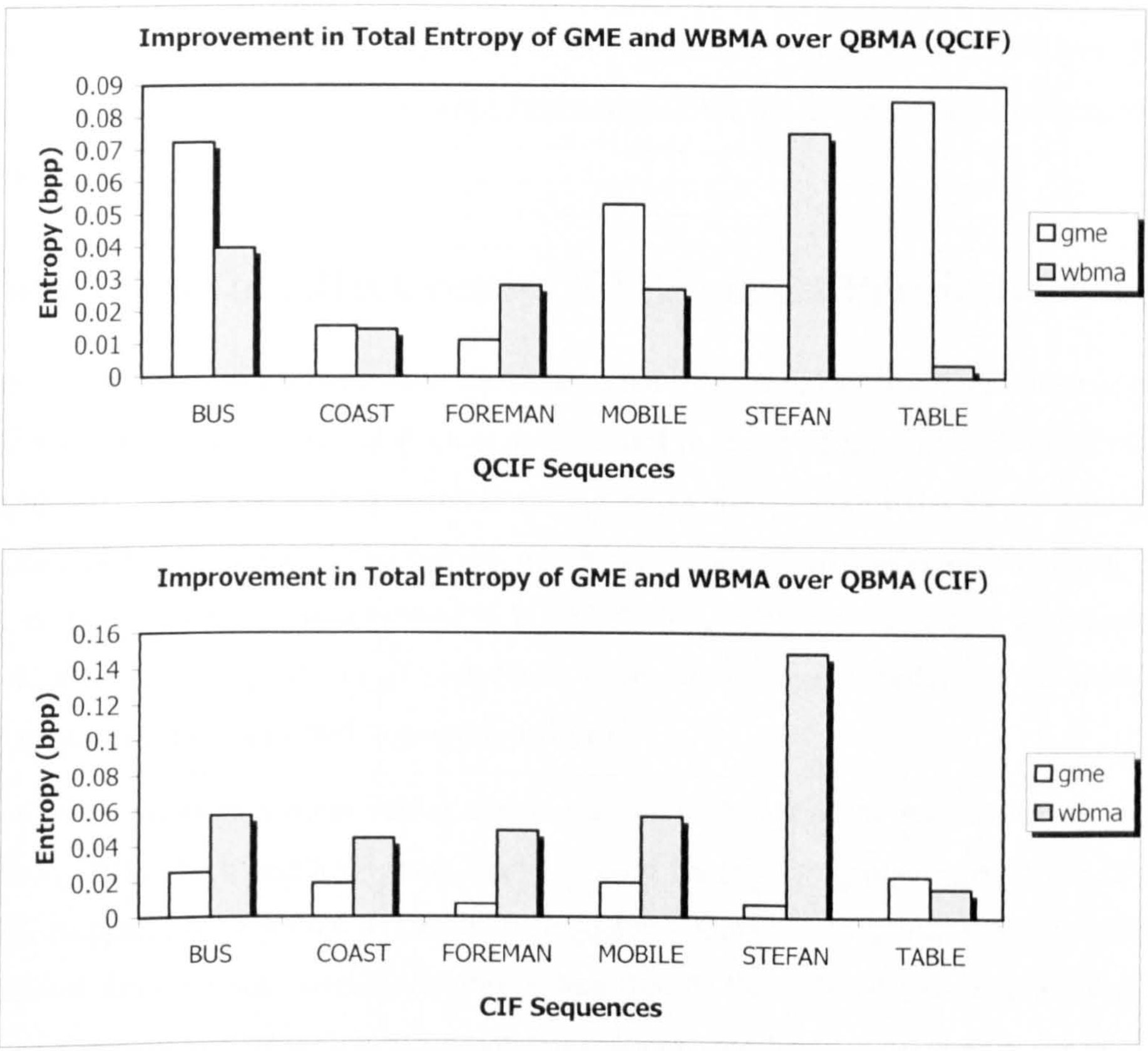


Figure 5.21. Charts showing the improvement in combined entropies of the by two GME related coding schemes over QBMA. See text below for the description of gme and wbma. The top chart is the results of QCIF sequences; the bottom chart shows the results of the CIF sequences.

Figure 5.21 shows the results of the two schemes:

1. gme – representation of QBMA vector field with the global affine parameter set (from SIRGME) and residual motion vectors.

2. wbma – two-pass QBMA, second pass performed on the warped reference frame with global motion parameters obtained from the first pass QBMA.

It can be seen that both gme and wbma outperforms the basic qbma. The wbma involving 2 QMBA, 1 SIRGME and 1 warping operations produces the least combined entropies for all QCIF and CIF sequences. The gme scheme marginally outperforms the basic qbma. The wbma scheme, on the other hand, provides much higher compression ratio at the expense of increasing complexity. The wbma coding system provides a better coding performance than the gme coding system in CIF sequences; the wbma coding systems does not appear to have as much success with the QCIF sequences. This is due to the fact that resolution in QCIF sequences is not fine enough to enable accurate extraction of individual segments. In summary, both wbma and gme can produce coding improvements of about 0.05 bits pixels per (bpp) in CIF@30fps and QCIF@10fps sequences This translate to around 1.2 kbits per QCIF frame and 4.8 kbits per CIF frame, or about 12 kbps improvement for the QCIF@10fps applications and 144 kbps savings for CIF@30fps

5.6 Comparison of Effectiveness GME verses Predictive Coding

In every video coding standard, motion vectors are coded differentially by a linear prediction scheme. For each block, a predictor is derived from certain central measure of a set of motion vectors from its causal neighbours. The actual motion vector is subtracted by this predictor and the difference vector is entropy coded and transmitted. This section investigates the effectiveness of the linear predictive coding for motion vectors commonly used in H.263, H.264, MPEG-1, -2 and -4 in reducing motion entropy, which derives the predictor of each block from evaluating the median of the motion vectors from the 3 neighbouring blocks (left, top and top-right).

The entropy of the different motion field is then compared with that of the residual motion vector field obtained from GME (the translation+zoom model is used for this comparison). Figure 5.22 shows the result of this comparison for both CIF and QCIF sequences. Contrary to intuition, the motion entropies of the predicted field are not consistently lower than that of the original vector field. That is, linear prediction of the motion vector field does not bring about significant reduction in entropy. In some cases, like COAST.QCIF and HALL.CIF, the entropies of the predicted difference vector fields are higher than the original field. This can be attributed to the fact that although motion vectors of neighbouring blocks may be correlated, the correlation is not effectively decoupled by the simple process of predictive coding. However, this does not mean that predictive coding of motion vector is not desirable. In fact, the difference vectors produced by linear prediction tend to have smaller x- and y- components. This simplifies the entropy code design; any variable length code which favours smaller value will suffice.

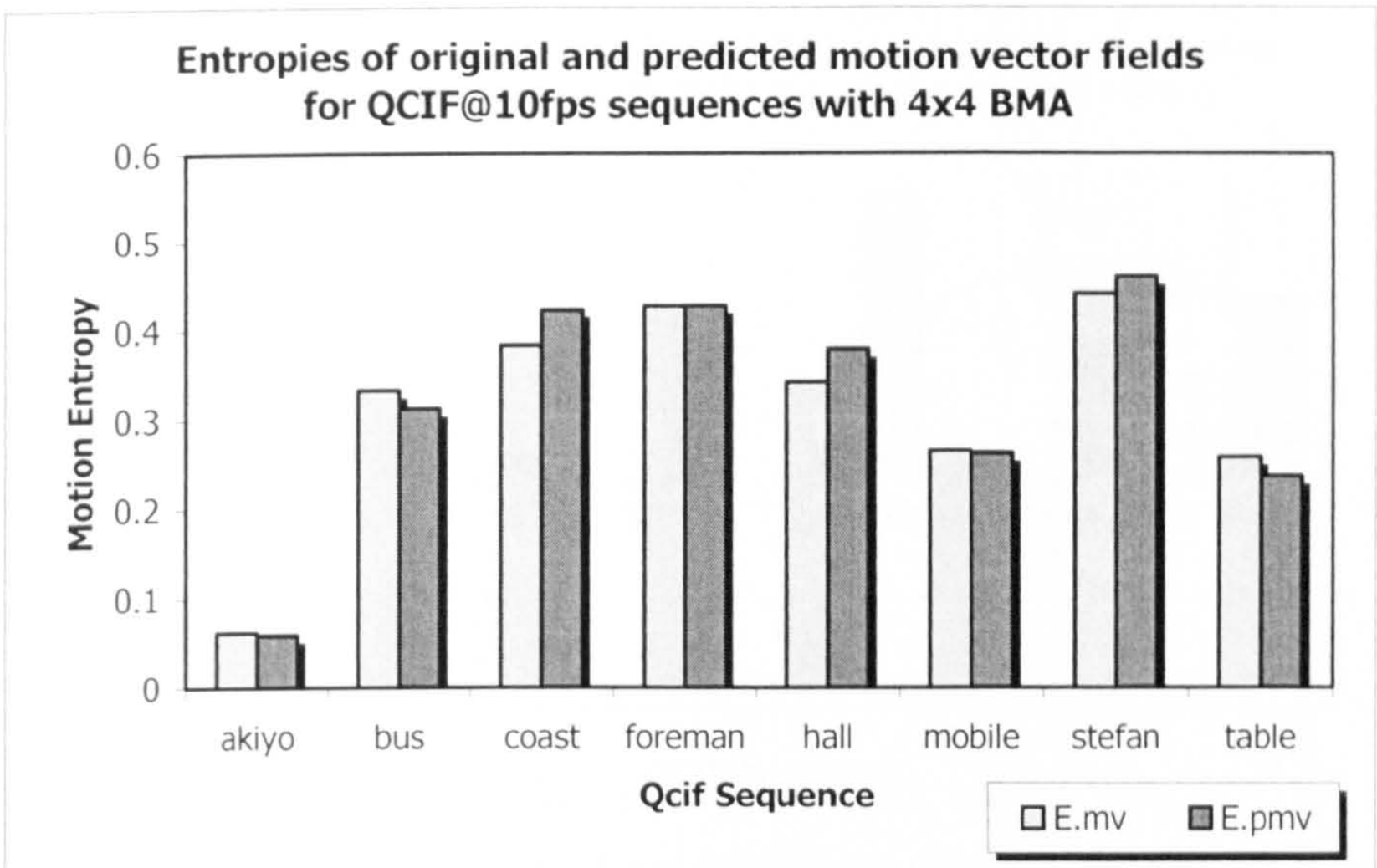
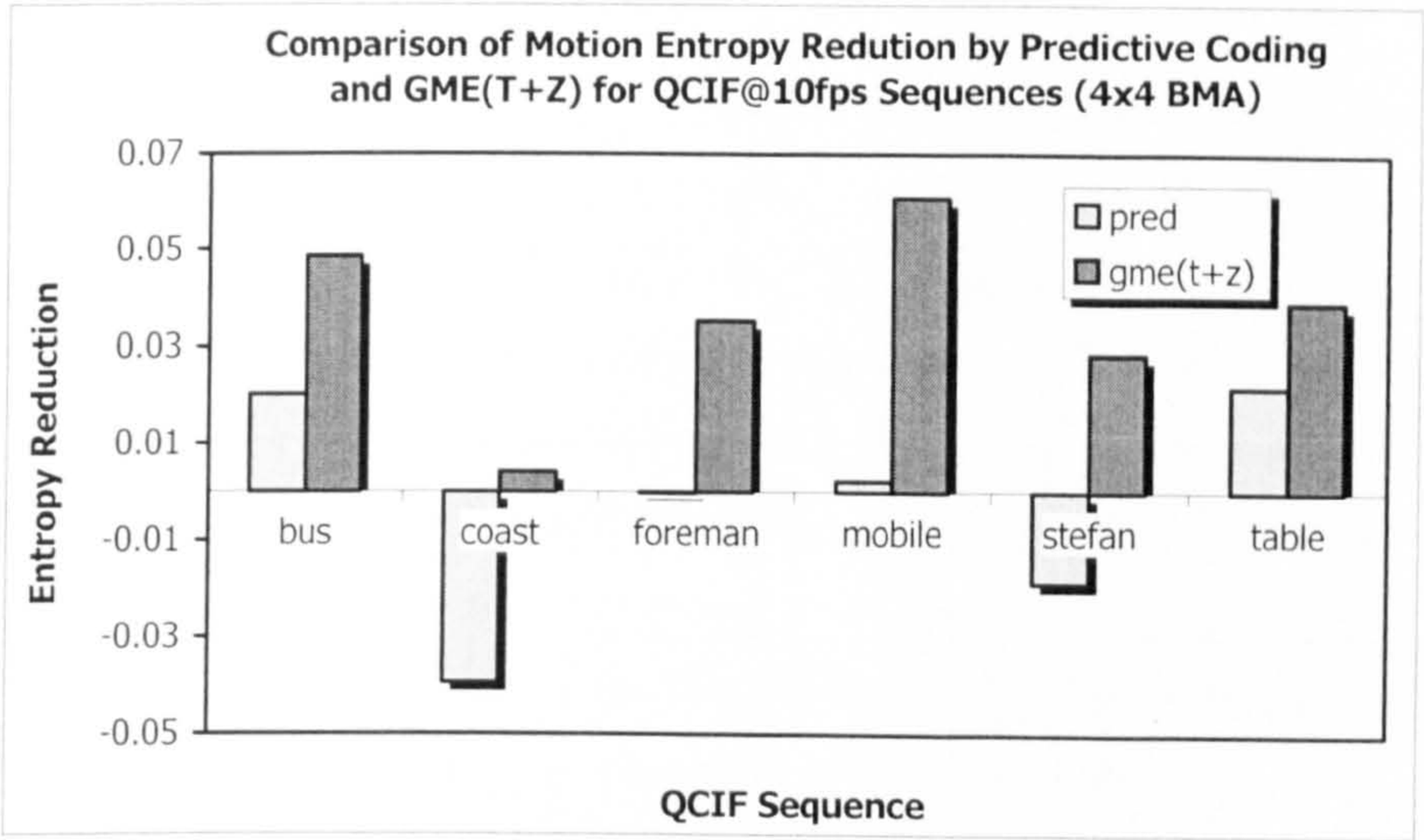


Figure 5.22. Charts comparing the entropies of original motion vector field (E.mv) and the difference field from predictive coding (E.pmv). Top chart shows the results of Qcif@10fps sequences and bottom chart shows those of Cif@30fps.

Next, the effectiveness of GME in reducing motion entropy in compared with that of predictive coding. Figure 5.23 compares the amount of entropy a simple translation+zoom GME can remove with that of predictive coding. GME consistently out-performs predictive coding for all the tested CIF and QCIF sequences. It can be concluded that GME can decouple the spatial correlation of the motion vectors much more effectively than simple linear prediction can.



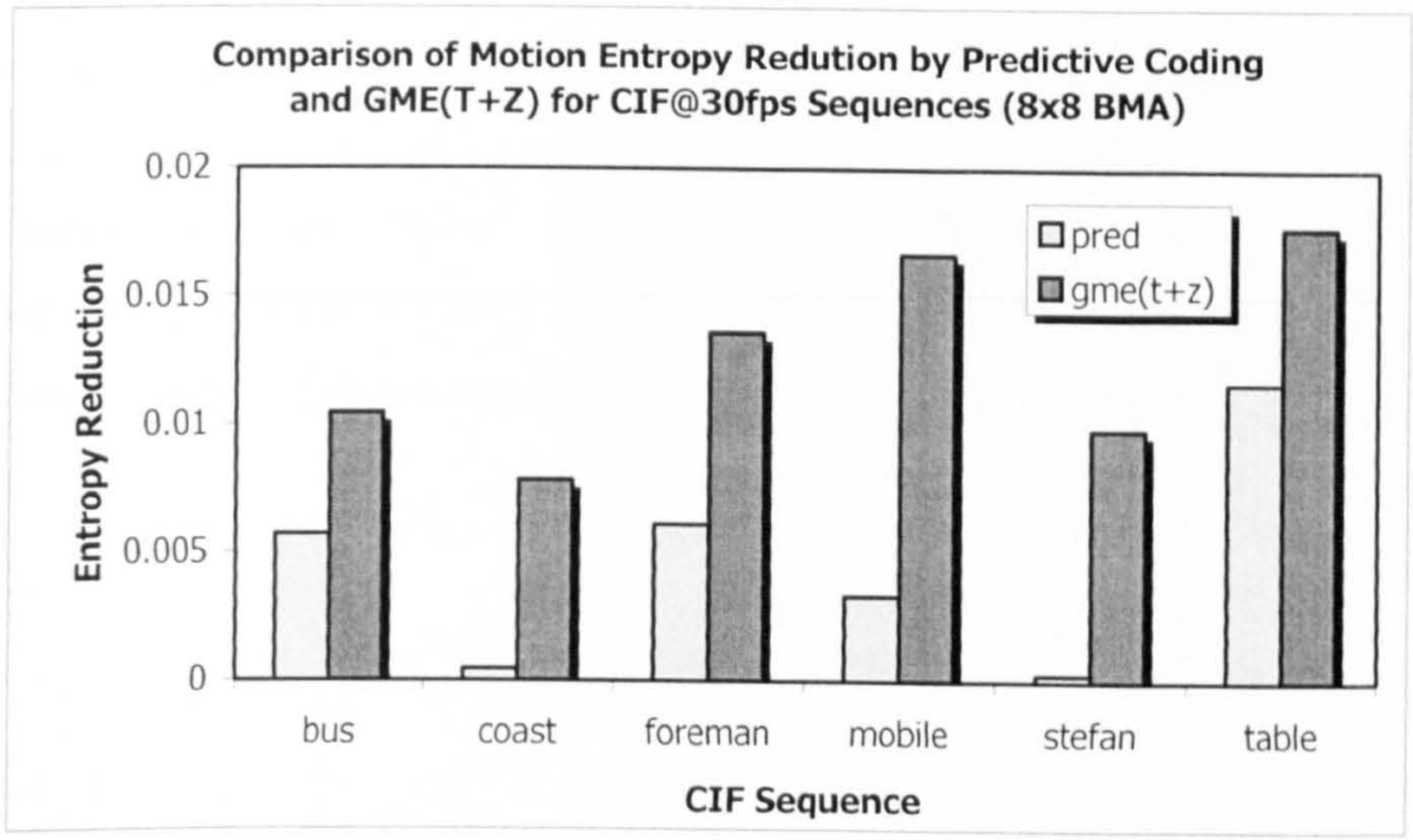


Figure 5.23. Charts comparing the entropy reduction capabilities of predictive coding (pred) and global motion estimation (gme); with gme, the translation+zoom (t+z) model is used. Top chart shows the results of QCIF@10fps sequences and bottom chart shows those of the CIF@30fps sequences.

5.7 Conclusions and Recommendations

Global motion estimation (GME) provides an improvement to the compression capability of local motion estimation methods. The amount of improvement is content-dependent, but in general, scenes with a dominant moving object or camera movement benefit most from GME. GME improves video compression by (i) compacting the local motion field into a single set of global motion parameters which requires very little overhead, and a residual motion vector field which contains less high-magnitude vectors; (ii) allowing a warped version of the reference frame to be used for local motion

prediction, which is bettered matched to the input frame than the original reference frame. Simulations show that global motion estimation provides a better compression ratio then local motion estimation.

Amongst the methods of GME, the regression methods based on an initial motion vector field is the most tractable computationally. This method is adopted into the existing QBMA architecture and an iterative algorithm, which results in the proposed SAD-map-based Iterative Regression GME (SIRGME). The initial field produces by QBMA is a smoother version of the traditional full-search BMA, and is more ideal for the regression-based GME. The initial robust statistics is based on the motion candidacy spread (MCS) used in QBMA, which has been shown to be a superior regression weights. The initial regression results are then iterated with Tukey's biweight M-estimator to arrive at the refined global parameter sets. Comparison with the traditional regression-based GME shows the SIRGME is more superior in producing a more compact motion vector field.

Although SIRGME is shown to provide a more compact vector field, the method does not perform optimally when the scene contains a substantial number of moving objects and/or a few moving objects occupying a relatively large area. In such instances, a more robust method is required to obtain the actual global motion parameters. The next chapter is devoted to the discussion of such a method, the Hough transform. A novel algorithm will be proposed which reduces tremendously the computational and memory requirements, which has potential in real-time and mobile video applications.

Chapter 6:

BMA-Based GME using Hough Transform

The Hough transform (HT) is one of the oldest robust transforms used in image analysis and computer vision. It has been used primarily to extract simple shapes from images. Essentially, the Hough Transform maps observation data from its original space into an appropriately quantized parameter space, and locates the most likely parameter values to through polling. Due to its robustness against outliers, the Hough transform has been used extensively for feature extraction and other image analysis in off-line processing. However, its heavy computation and large memory requirements have prevented it from being used extensively in real-time applications.

In video compression, global motion estimation brings about significant coding gain in sequences with camera-induced apparent motion. Traditional iterative regression and gradient methods based on a dense optical flow field or on intensity conservation principles are computationally intensive, and do not perform well in presence of foreground objects moving independently. Aperture and occlusion problems degrade their performances further; finally, inappropriate initial conditions would sway these iterative algorithms towards local minimums.

In view of the robustness of Hough transform against noise, it is very suitable for global motion estimation. Previous attempts [Adi-95][Bob-93][Kal-96] yielded favourable results, but were either based on a dense optical field or on matching a set of candidate points between two frames. These methods are unsuitable for video coding systems. In this thesis, a novel Hough Transform based global motion estimation algorithm is proposed. It relies on QBMA to obtain a globally-smooth motion vector field, followed by a Hough transform-based Global Motion Estimation (HGME) on the field. The side information from QBMA is the reliability measure of motion vector for each block (termed MCS in the previous chapter) which is used as a polling weight in the Hough space. In the proposed method, the Queue-based BMA provides a favourable sparse vector field in the form of a set of candidate points for polling while the Hough transform provides the robustness to outliers for the subsequent GME. The QBMA produces a motion vector field which is smooth within texture-less regions, yet preserves discontinuities at boundary blocks; the reliability measure reduces the effect of previously-occluded blocks on HGME. Using the proposed method, an accurate estimation of the global motion parameters can be produced with relatively low processor and memory requirements, which makes it suitable for real-time implementation.

6.1 Introduction to Hough Transform

In this section, the basic principles behind the common problem of line detection are described; the process of extending the technique to the problem of global motion estimation is laid down. In the classical shape detection application, the classical Hough Transform maps edge points that potentially lie on the shapes into the shape's parameter space, which is appropriately quantized. The shapes are then identified and located through polling by counting how many edge points pass through the particular shape. The classic example is the detection of straight lines given a set of edge points. Figure 6.1 illustrates the basic components required for detecting the simplest form of shape, the straight lines.

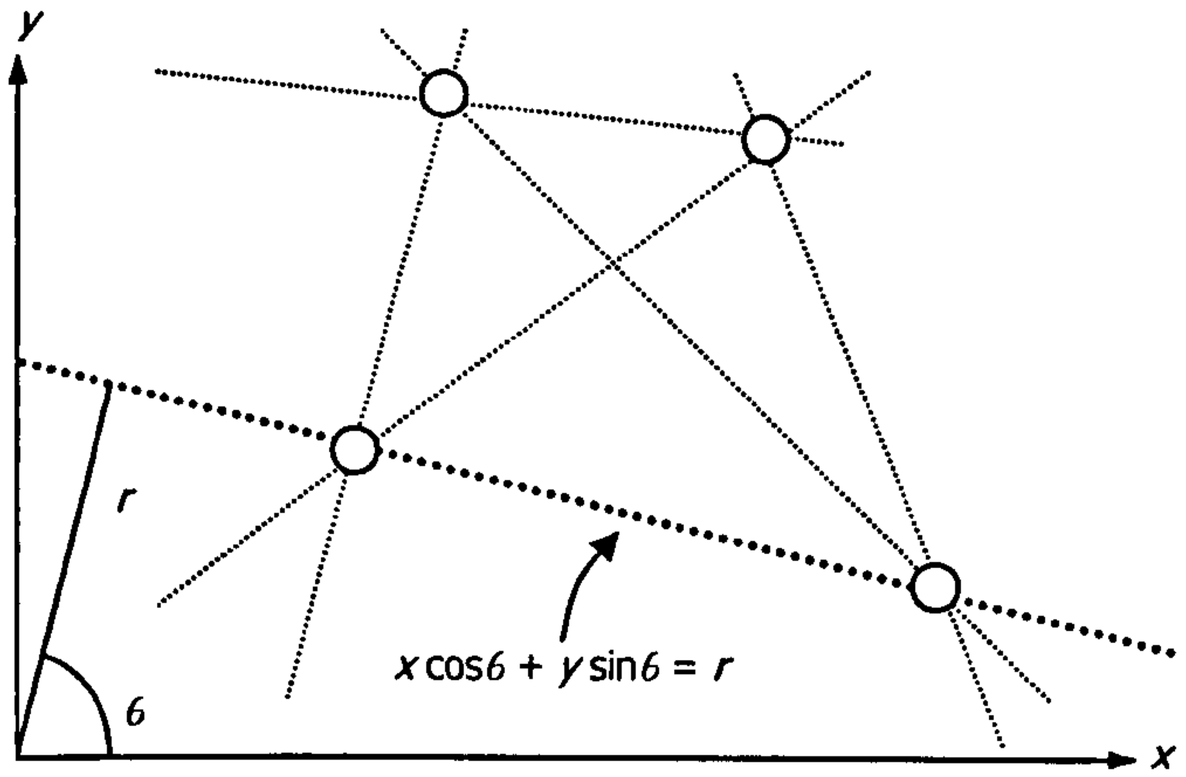
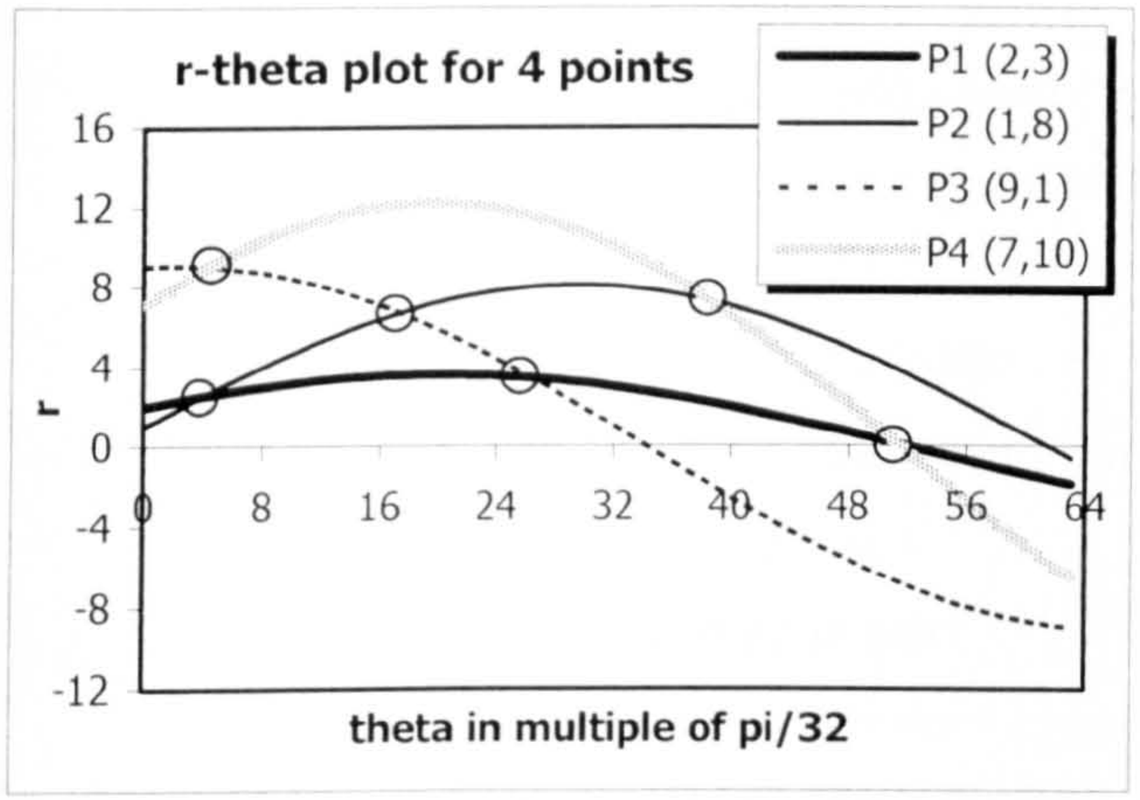


Figure 6.1. A figure showing the essential elements of Hough Transform-based line detection. The figure shows 4 edge points. 6 lines can be detected, each passing through 2 points. The lines are represented as the equation shown, with the parameters r and θ.

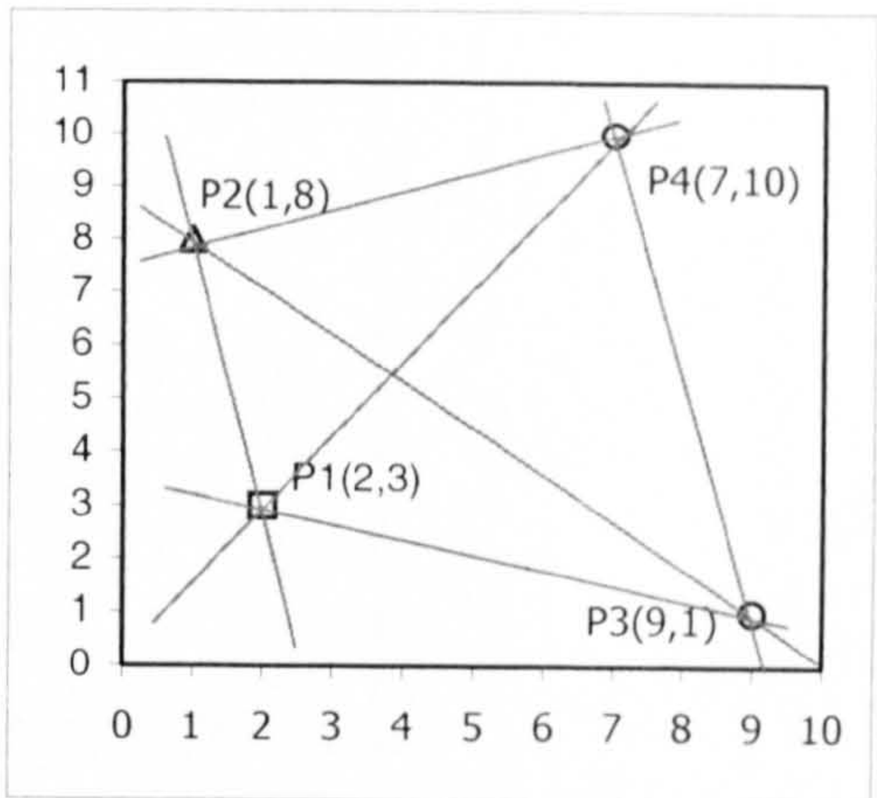
Given a set of four edge points, we want to find the possible lines (edges) on which the points may lie. For this purpose, a straight line is defined as:

$$x\cos\theta + y\sin\theta = r \qquad \text{Eq 6-1}$$

The (x, y) pair is the co-ordinate of the point through which the line passes; r is the length of a normal from the origin to this line and θ is the orientation of r with respect to the x-axis. In an image analysis context, the coordinates of the points of edge segments (x_i, y_i) are known and (r, θ) are considered variables. Each point (x, y) forms a curve on the (r, θ) plane.



(a)



(b)

Figure 6.2 Illustration of edge finding by Hough transform. (a): (r, θ) plot of four points. (b): Co-ordinates of the four points. There are six intersections in the left chart (identified by the circles), which correspond to the six lines formed by each pair of points.

We can see from Figure 6.1 that six possible lines can be formed amongst six possible pairs from the set of four points. This corresponds to the six intersection points in Figure 6.2 (a), whose (r, θ) co-ordinates indicate the equation of the lines. Edge points lying along a line would correspond to a point in the (r, θ) plane where many curves intersect. In order to detect the points of intersection, the classical Hough Transform is implemented by quantizing the Hough space into finite intervals and each interval forms an accumulator bin. As each edge points (x_i, y_i) is parsed, the accumulator bins in the form of $[r, \theta]_j$ though which the curve of the equation:

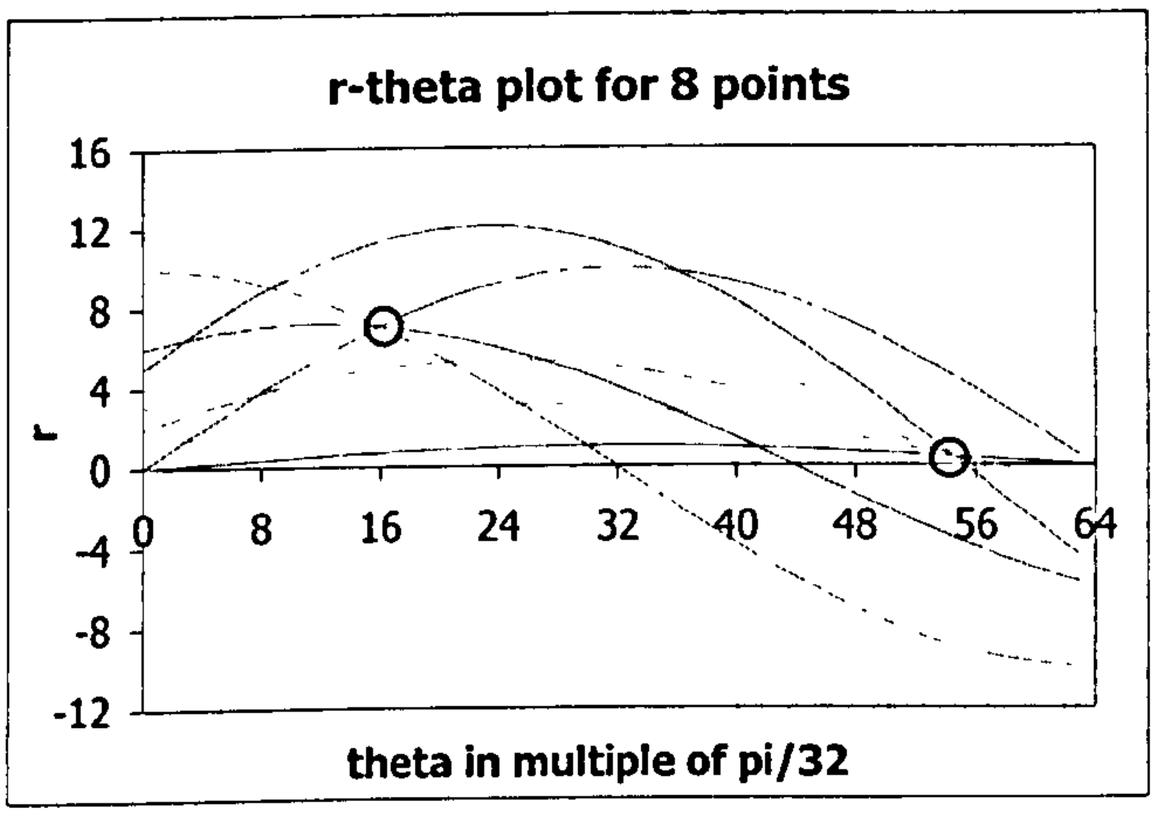
$$x_i \cos\theta + y_i \sin\theta = r \qquad \text{Eq 6-2}$$

passes through. After all the points are parsed, lines in the images are identified as peaks in the accumulator bins. The classic algorithm can be summarized in the following pseudo code:

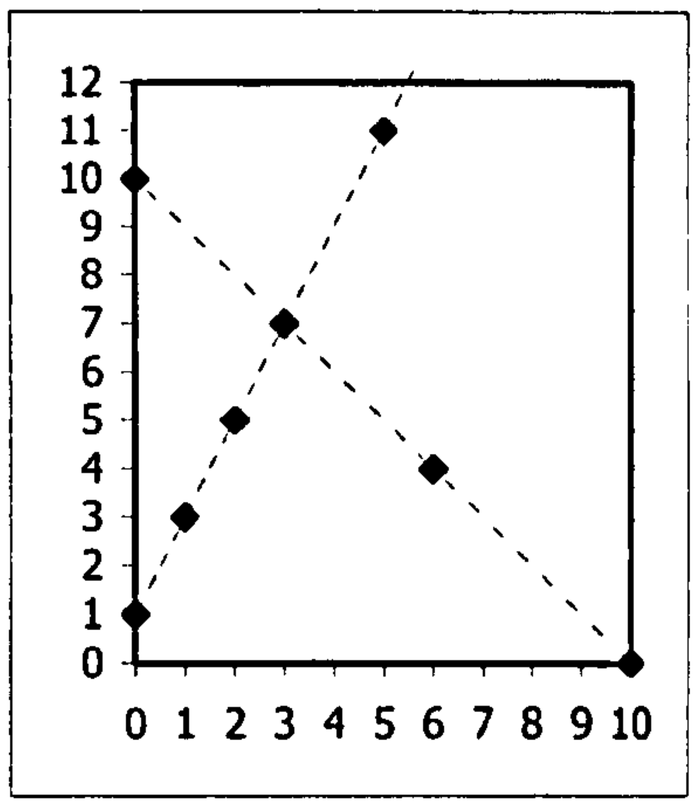
```
For each edge point (x_i, y_i),
    For each quantized value for θ, θ_m
        r = x_i cosθ_m + y_i sinθ_m
        Quantize r to r_n
        Increment Bin [θ_m][ r_n]
    Next θ_m
Next edge point
```

Figure 6.3 Pseudo code of the class line detection by Hough Transform

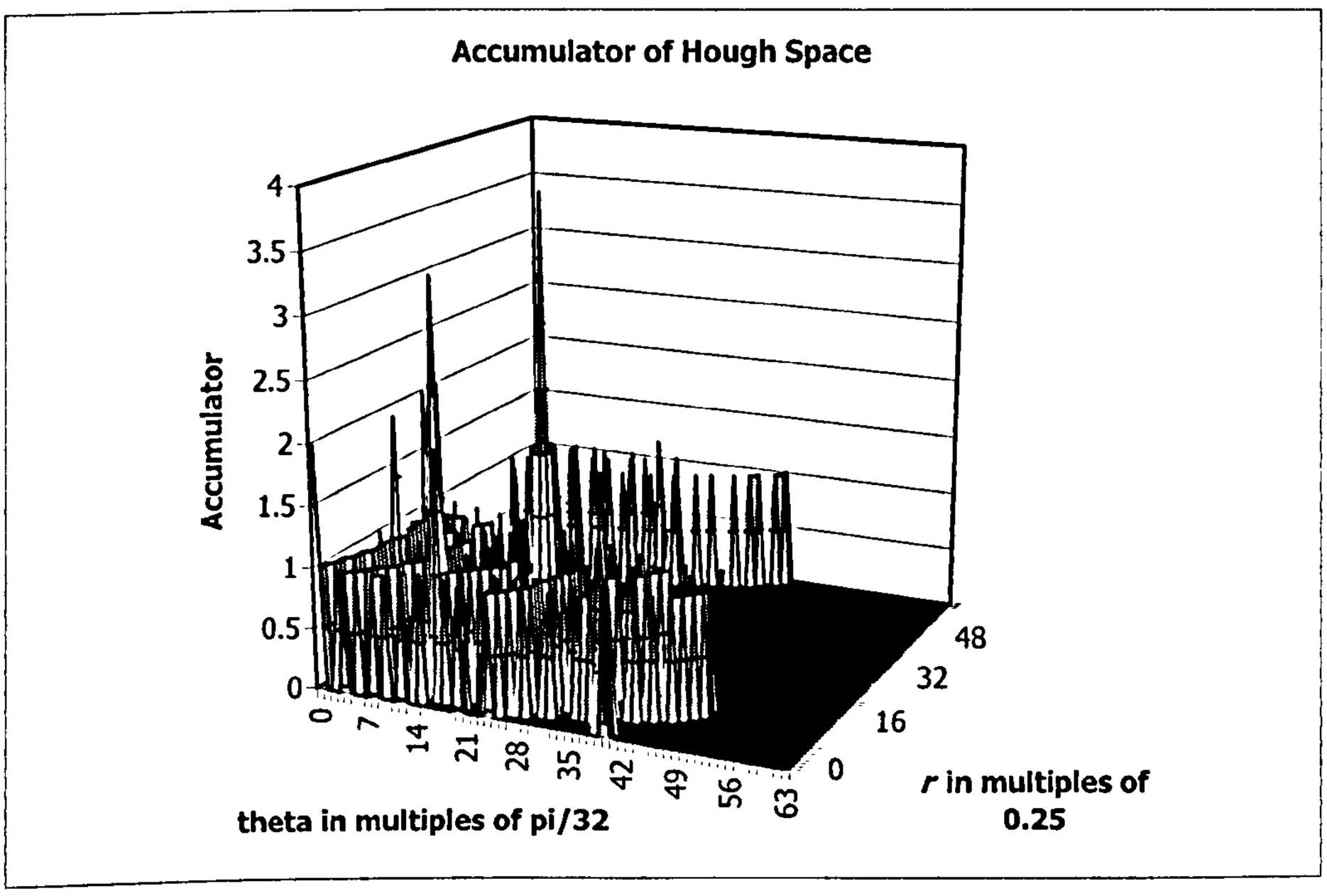
As an illustration of Figure 6.3, Figure 6.4 shows the two lines formed by joining the 8 observation points in Figure 6.4 (b), which manifest themselves as the two points of intersection of the Hough curves in Figure 6.4 (a). Figure 6.4 (c) shows the accumulator bins after the Hough Transform, both r and θ are quantized into 64 bins of ranges 0-16 and 0-2π respectively. The two peaks are distinct which corresponds to the two lines detected. A few problems can be seen from Figure 6.4 (c). Firstly, some accumulators show minor peaks of value 2, which corresponds to lines joining pairs of points other than the 2 major lines. In real applications, the distinction between major peaks and peaks due to noise are sometimes not very obvious. In some cases, the number of features to extract is unknown and has to be determined through some threshold value set by some other means. Secondly, the values of the two peaks are 3 and 4, which is less than the number of points that passes through the lines (4 and 5 respectively). This is due to the quantization of the Hough space, which fails to keep all the intersection points into a precise location in the Hough space.



(a)



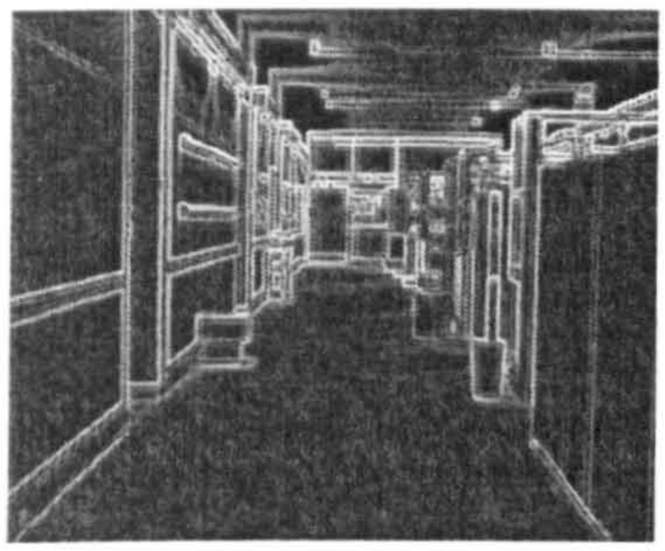
(b)



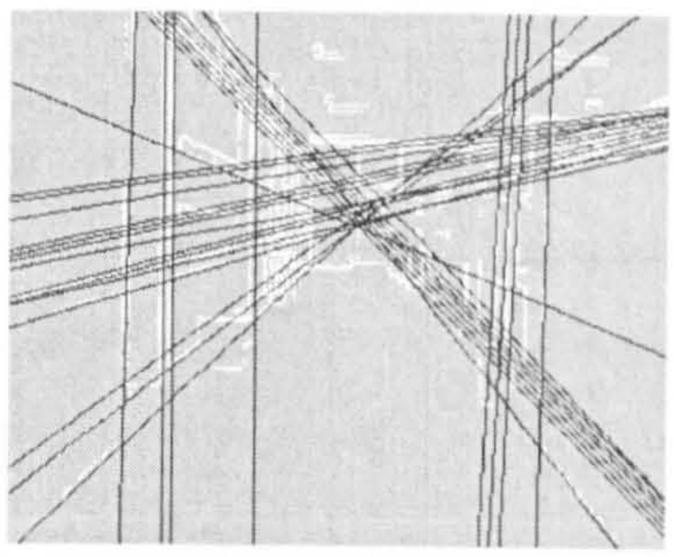
(c)

Figure 6.4 Another illustration of edge finding by Hough transforms. (a): (r, θ) curves of eight points. (b): Co-ordinates of the eight points. There are two major intersections in the left chart, which correspond to the two lines each formed by more than three points. . (c): The accumulator of Hough space. Two major peaks can be detected, which corresponds to the two lines.

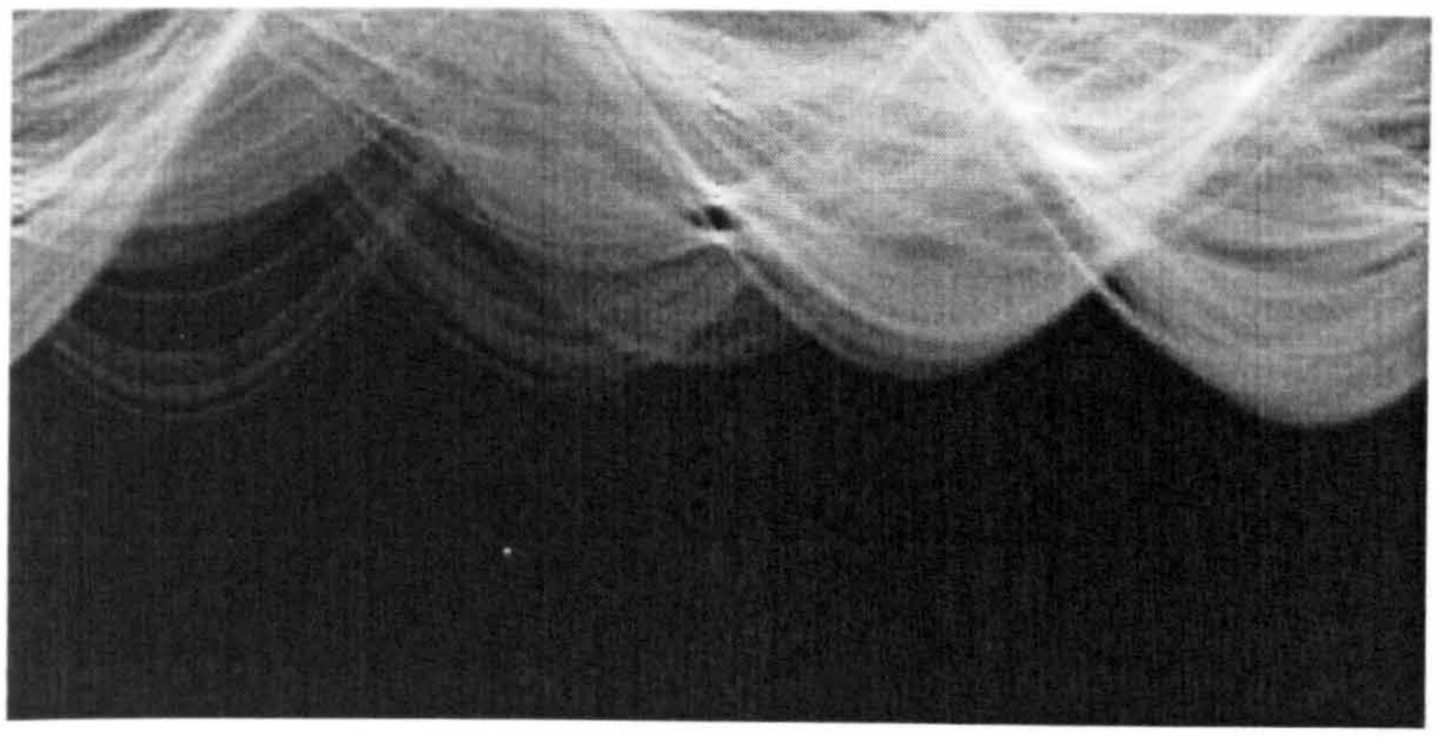
As an illustration, the Hough transform is applied to find edges in a frame from the HALL.CIF. The edge diagram in Figure 6.5 (a) is found by Canny edge detector, which is used as the input to a Hough transform with 256×128 cells with the range $[0, 2\pi) \times [0, 352)$. Edges found in Figure 6.5 (b) match those in Figure 6.5 (a).



(a)



(b)



(c)

Figure 6.5 Illustration of edge detection by Hough transform. (a) edge map detected by canny edge detector; (b) lines detected by Hough transform; (c) Hough space (r, θ) of the edge map.

Care has to be taken to select the quantization steps in the $(r,6)$ space. When the bins are too finely quantized, an intersection point of several sinusoids may end up spreading into a few bins; conversely, when the bins are too roughly quantized, the detected lines cannot be located precisely.

Theoretically, the Hough transform follows the principles of maximum likelihood estimation. For a certain range of quantized Hough space, each (x, y) is mapped into the $(r,6)$ space and the points that map into the locations in the Hough space are accumulated. The accumulator can be considered as a 2-D histogram. The relationship between the parameter vector $(r,6)$ and an observation data (x, y) can then be made by assuming that the data set $\{(x, y)\}$ represents the complete sample of the probability density function (pdf) of $f(r,6)$, and then we have:

$$L[(r,\theta)|(x, y)] = P[(x, y)|(r,\theta)] \qquad \text{Eq 6-3}$$

By counting each accumulated bin, we are determining which bin produces the highest relative frequency, giving an estimate of $L[(r,\theta](x,y)]$. Because of its global search nature, the Hough transform is robust, even when there are a high percentage of errors in the data. For better accuracy in locating the solution, we can increase the resolution of the parameters space. However, the size of the accumulator increases rapidly with the required accuracy and the number of unknowns. This technique can be readily adopted to solve for the parameters of a global motion model.

In some cases, the edge points are not binary value; a greyscale image when each point (x, y) is represented as $g(x, y) \in [0,1]$. The greyscale value is proportional to the likelihood of that point being an edge point. In such case, the accumulator is incremented by the variable value $g(x, y)$ instead of unity. In this form, the Hough transform is basically the discrete version of the Radon transform, typically used for reconstruction of three-dimensional images from two-dimensional projections, represented as:

$$R(r,\theta) = \int_x \int_y g(x,y)\delta(x\cos\theta + y\sin\theta - r)dxdy \qquad \text{Eq 6-4}$$

Here, the delta function defines integration only over the line. As in the Hough transform, the Radon operator maps the spatial domain (x, y) to the projection domain (r,θ), in which each point corresponds to a straight line in the spatial domain.

The above description illustrates the steps required to detect straight line segments from edges. More complicated shapes can be detected, with increasing complexity. For instance, to find circular arcs, we need the centres (x_c, y_c) and radii r of the circle, thus requiring a 3-dimensional Hough transform space. This is used intensively in biomedical applications to detect blood cells and to recognize other simple shapes in manufacturing applications. To identify arbitrary shapes, we need the generalized Hough transform, described analytically as

$$H(\Omega) = \sum_{i=1}^{N} p(\mathbf{X}_i, \Omega) \qquad \text{Eq 6-5}$$

$$p(\mathbf{X},\Omega) = \begin{cases} 1 & \forall(\mathbf{X},\Omega) : \{\Lambda : f(\mathbf{X},\Lambda) = 0\} \cap C_\Omega \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

In Eq 6-5, $H(\Omega)$ is the Hough transform where Ω is the parameter set whose value is to be estimated and $\{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_K\}$ are sets of observation points, C_Ω is a finite-sized cell centred at the point Ω in the parameter space, and $f(\mathbf{X}, \Lambda) = 0$ is the parametric constraint or the relation between the observation points $\mathbf{X}$ and the parameter space Λ. In the remaining of this thesis, Λ shall be referred to as the Hough

space while $\mathbf{X}$ the observation space. For a straight line fitting, $\Omega = (r, \theta)$, $\mathbf{X} = (x, y)$ of an edge point, C_Ω is the partition (quantization) of the Hough space, and $f(\cdot) = r - x\cos\theta - y\sin\theta$.

6.2 Hough Transform-based GME (HGME)

The advantages of using the Hough Transform for global motion estimation are obvious when we consider its robustness compared with other methods. A simple illustration in Figure 6.6 demonstrates this. We use a typical frame from the COAST.QCIF sequence where there are two dominant motions. The major motion is due to the panning background while the minor motion is due to the ship, as shown in Figure 6.7. For simplicity and graphical presentation, the translational motion model is used:

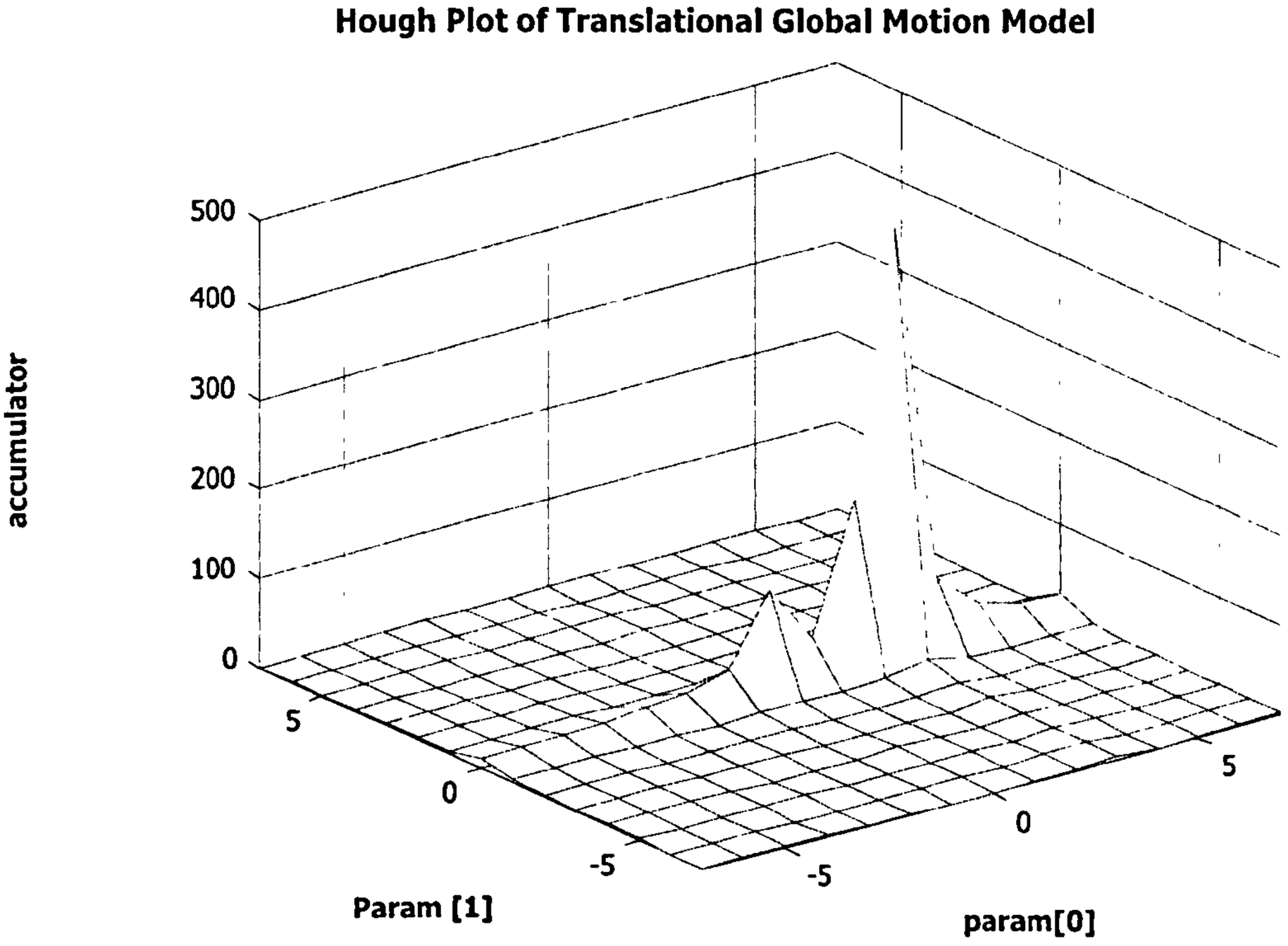
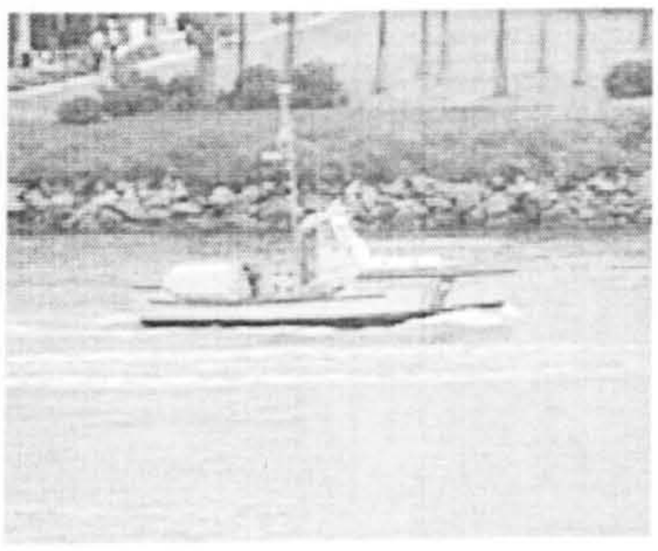


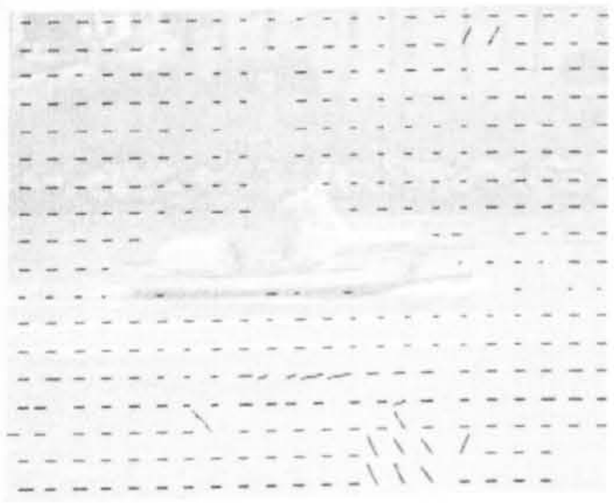
Figure 6.6 A plot of Hough accumulators using the translational motion model of one frame from the COAST.QCIF sequence.

As is evident in Figure 6.6, the Hough transform can spot the global motion component (the higher peak) as well as some minor moving objects (lower peak) in the Hough space. A close inspection of the contour plot in Figure 6.7 reveals that the Hough Transform is more robust to noise (in this case the motion due to the small object) than the regressive method. The cross is the solution found by the regression.

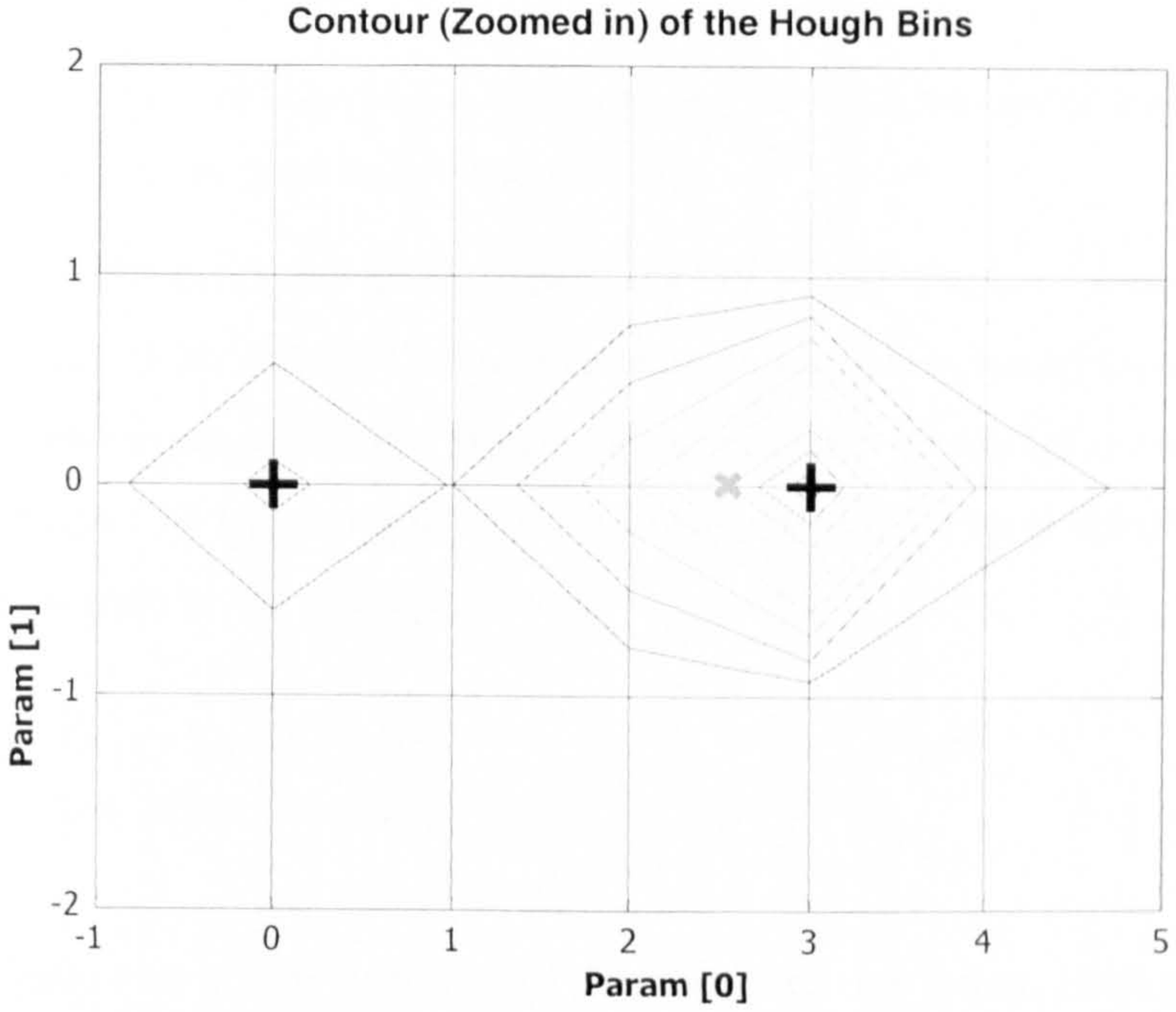
Essentially, Hough transform is different from all optimization methods in one crucial aspect. Hough transform locates the desired parameters from majority polling; whereas other methods find solutions in the parameter space which optimize certain criteria, which will inevitably be influenced by observations points which do not arise from the true parameters. This makes Hough transform an ideal algorithm to extract global motion parameters from scenes containing different foreground objects moving independently, as long as global motion is the major contributor of the apparent motion. In the remaining sections, the Hough-based global motion estimation introduced in thesis will be referred to as HGME.



(a)



(b)



(c)

Figure 6.7 Hough Transform result of frame 201 of COAST.QCIF. (a): The frame; (b) the map motion vector map used for Hough Transform. (c) Contour plot of the same Hough transform for Translational global motion parameters. The two "+"s indicated the 2 detected global motion (the major motion is the background while the minor is due to the ship. The "×" indicates the motion parameters found by iterative regression.

6.3 Models, Extent and Resolution

Let us now look at how the Hough transform, as described in Eq 6-5, can be used in global motion estimation. We use the 3-parameter zoom-translation and the 6-parameter affine models for illustration purposes:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} a_0 & 0 \\ 0 & a_0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} \Rightarrow \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} x & 1 & 0 \\ y & 0 & 1 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} \quad \text{Eq 6-6}$$

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} a_0 & a_1 \\ a_2 & a_3 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} a_4 \\ a_5 \end{bmatrix} \Rightarrow \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} x & y & 0 & 0 & 1 & 0 \\ 0 & 0 & x & y & 0 & 1 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{bmatrix}$$

The Hough space would be $\Omega_3 = \{a_0, a_1, a_2\}$ and $\Omega_6 = \{a_0, a_1, a_2, a_3, a_4, a_5\}$ for the 3-parameter and 6-parameter models respectively. The observation space is $\mathbf{X} = \{(u, v, x, y)\}$.

As in previous chapters, simulations in this section use QCIF sequences (176 × 144) with 8 × 8 blocks. Simulation results in Figure 5.15 to Figure 5.20 show that the scaling and rotating parameters of Ω_3 (a_0) and Ω_6 (a_0, a_1, a_2, and a_3) all lie within the range of ±1/4. Hence the extent of the Hough parameters space should also lie within similar ranges. To have some idea of the lowest meaningful resolution, we look at the sensitivity of the values of u and v to the parameters in Ω_6.

$$\begin{array}{lll} \Delta u = x\Delta a_0 & \Delta u = y\Delta a_1 & \Delta u = \Delta a_4 \\ \Delta v = x\Delta a_1 & \Delta v = y\Delta a_3 & \Delta v = \Delta a_5 \end{array} \quad \text{Eq 6-7}$$

With QBMA, motion vectors are estimated to the accuracy of integral pixel resolution, giving a tolerance of $\frac{1}{2}$-pixel for Δu and Δv. In order to ensure that at least half the blocks would produce non-zero motion vectors Δa_0, Δa_1 Δa_2 and Δa_3 should not be smaller than Δr as indicated in Eq 6-8

$$\frac{1}{2} < \frac{1}{\sqrt{2}} \min\left(\frac{176}{2}, \frac{144}{2}\right) \Delta r \quad \text{Eq 6-8}$$
$$\Delta r > 0.01$$

We shall choose the resolution of Δa_0, Δa_1 Δa_2 and Δa_3 to be $\frac{1}{64}$, Thus arriving at a dynamic scale of 1:64. Similar dynamic scale for Δa_4 and Δa_5 are used ($\frac{1}{2}-16$ pixels). Similar rationale can be used to derive the accuracy of Ω_3.

In spite of its immunity to the effects of outliers, the Hough transform suffers from two major shortcomings. Firstly, the Hough space is quantized, which reduces the accuracy of the parameters. Naturally, the accuracies can be improved by using smaller quantization steps. Secondly, doing so will

increase the memory and processor requirements many-folds. Consider the 6-parameter Hough space; doubling the resolution requires a $2^6 = 64$ times increase in memory requirements for accumulator storage, and possibly as many times the clock cycles for processing. Hence the problem is a matter of comprise between accuracy and processor/memory requirements. The HGME algorithm proposed in this thesis has 3 novelties which improve accuracy without the expense of increased processor/memory requirements. The following sections describe each improvement in detail.

6.4 Novel Approaches to HGME

6.4.1 Sub-Bin Peak Location Refinement

Global motion estimation by Hough transform identifies global motion parameters from the location of the fullest accumulator bin. Usually, the peak location where the maximum value occurs, $\{a_0, a_1, \cdots\}^*$, is the centroid of the bin. In cases where there are too few bins, the estimated parameters may be too heavily quantized to be useful. In sub-bin peak location refinement, we propose an improvement of each parameter a_j* by assuming that the parameters are mutually independent of each other and that the variation of the accumulator values can be approximated by a quadratic model within the vicinity of the peak bin.

As is shown in Figure 6.8, each parameter a_j^* can be refined to a_j^{**} by Eq 6-9 which is a continuous real value in the range $a_j^* - 0.5 \le a_j^{**} \le a_j^* + 0.5$.

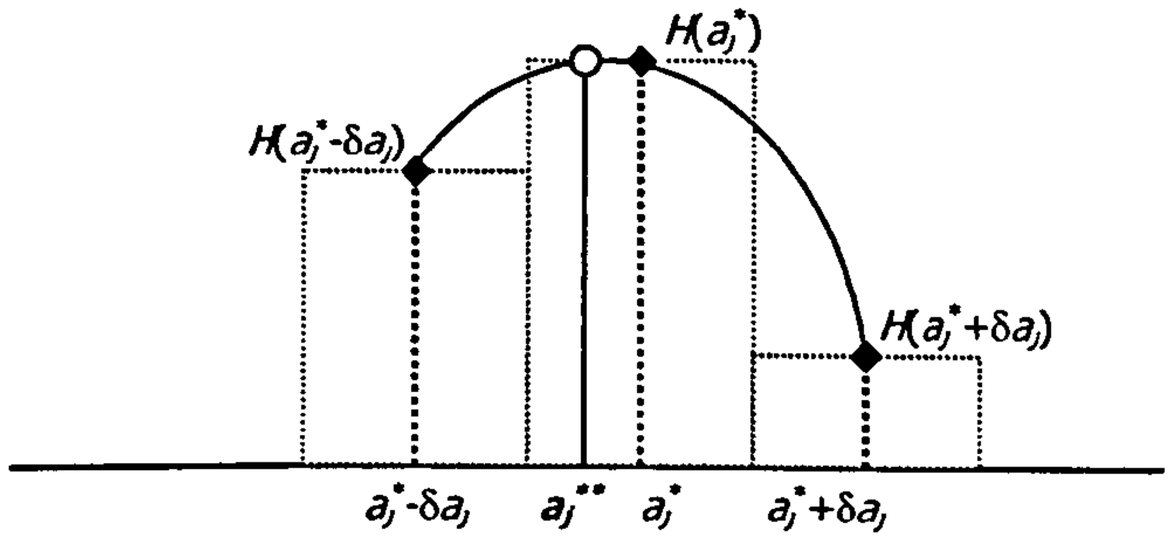


Figure 6.8. Quadratic model to improve a_j* estimate.

$$a_j^{**} = a_j^* - \frac{\left[H\left(a_j^* + \delta a_j\right) - H\left(a_j^* - \delta a_j\right)\right]\delta a_j^*}{H\left(a_j^* + \delta a_j\right) + H\left(a_j^* - \delta a_j\right) - 2H\left(a_j^*\right)} \qquad \text{Eq 6-9}$$

Another added advantage of sub-bin peak location refinement is related to the progressive resolution improvement described in the next section. Essentially, progressive resolution improvement applies an

additional iteration of the Hough transform within the peak bin. If the true peak lies as the boundary of the bin, progressive resolution improvement will still lie at the boundary of the refined space. By offsetting the centre point towards a better estimate of the peak location, the refined peak would be more likely situated around the centre of the subsequent Hough space.

6.4.2 Progressive resolution improvements

For each Hough space, we use a coarse-to-fine resolution approach, which is similar to the adaptive Hough transform [Tia-95]. As an illustration in Figure 6.9, a 2-D Hough space with 5x5 bins undergoing a two-pass refinement is equivalent to having 25x25 bins without incurring the added memory requirements. Both our 3-D and 6-D methods use the 2-pass refinement to reduce memory requirements.

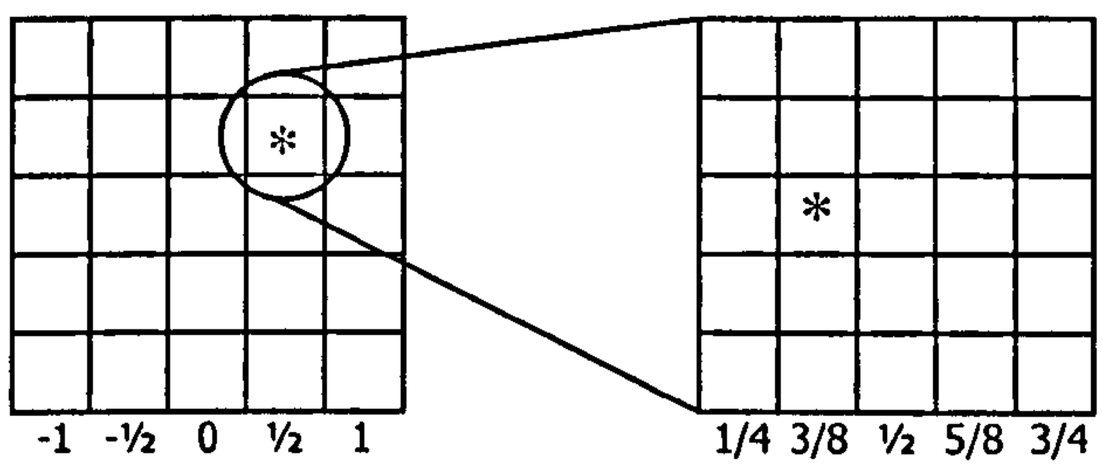


Figure 6.9. Illustration of resolution improvements.

In normal circumstances, applying progressive resolution alone would be sufficient to resolve the location of a parameter at a higher accuracy; this is not the case when the case when the true parameter lies right in between two bins. As illustrated by Figure 6.10, the true peak lies at the boundary of two bins in the original Hough space. By applying another Hough transform with smaller quantizer step centred at a_j^* does not locate the peak as it still lies at the boundary point and may be totally missed if the true peak lies just outside the second Hough space. By applying sub-bin peak location refinement (see right column of Figure 6.10), the subsequent Hough space is centred on the peak location and this peak can be accurately refined.

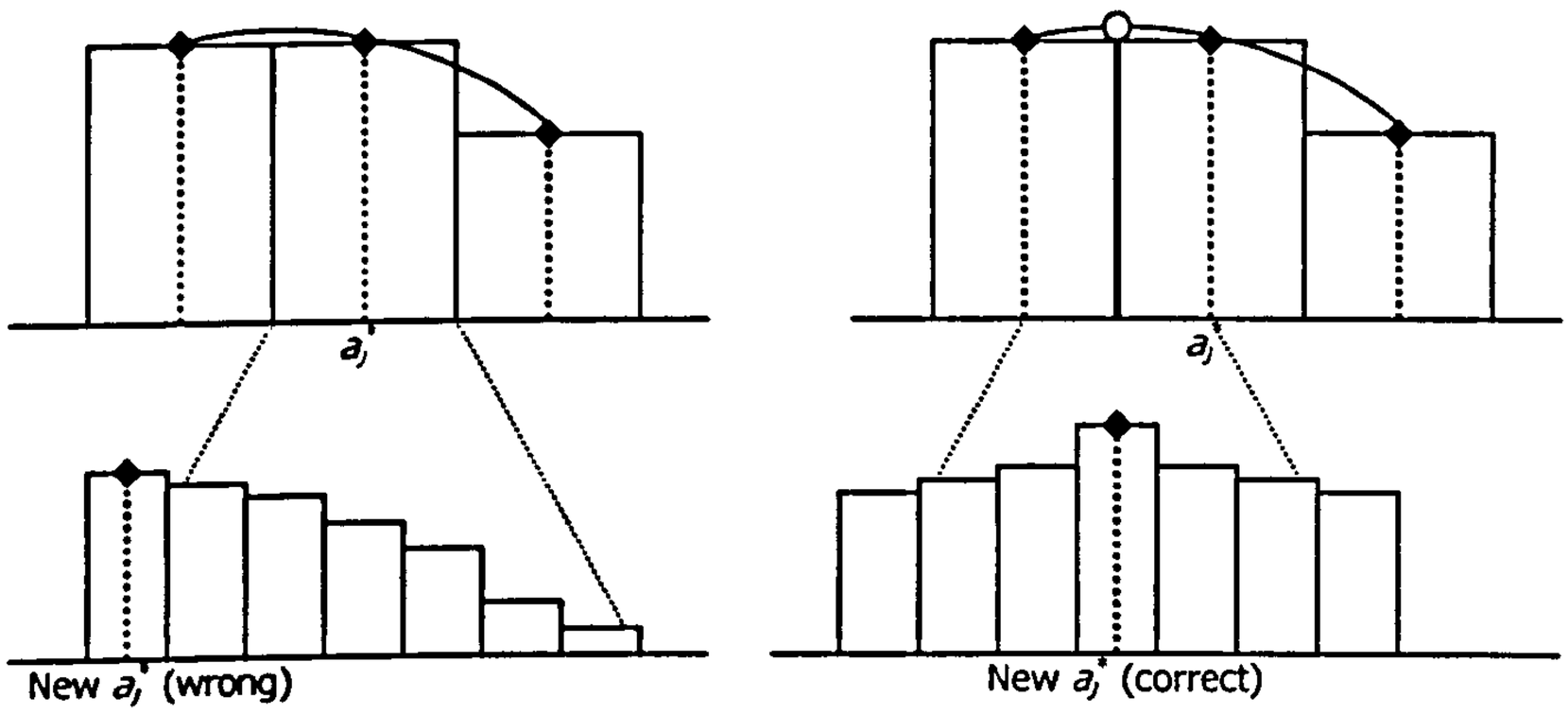


Figure 6.10. Illustration of how applying sub-resolution peak location prior to sub-progressive resolution improvement resolves the problem of boundary peak. Left: with only sub- progressive resolution, the boundary peak (new a_j^*) cannot be located correctly. Right, the centre of the new Hough space is offset to where the peak is most likely to occur (at the boundary two bins in the original Hough space); the subsequence Hough transform manages to locate the true peak.

6.4.3 Progressive model improvements

The 3-parameter model requires much less memory than the 6-parameter model. Furthermore, natural sequences are sufficiently modelled by the zoom + translation model (refer to the last section of previous chapter). Hence our algorithm uses a 2-step approach: a_0, a_4 and a_5 are estimated first (with $a_3 = a_0, a_1 = a_2 = 0$) using the 3-D Hough transform. Subsequently a 6-D Hough transform is performed to further improve the solution. This has been shown to reduce the complexity of the HGME vastly without compromising accuracy.

As an illustration, consider 129-bin ($[-64,64]$) accuracy for a 3-D and 6-D Hough transform. The 3-D case requires $129^3 \approx 2$ Mega bins; the 6-D transform requires a daunting $129^6 \approx 4.6$ Tera bins. Using state-of-art level of technology, the former can be implemented easily; the latter would pose a difficult challenge to even the most sophisticated machines in the years to come. By assuming that the zoom and translation parameters dominate the affine motion model, the HGME algorithm performs an initial 3-D Hough transform to locate the three dominant parameters. Subsequently, another Hough transform using the six-parameter model is centred on the three parameters found with the previous Hough transform and the assumed values of the remaining parameters. Assuming we use the same 2 Mega bins

of memory space from the 3-D Hough transform, we can perform an 11-bin 6-D Hough transform to achieve the same precision as the 129-bin 6-D transform.

Although high precision is achieved without incurring excessive memory costs, progressive model improvement is based on the assumption that the affine model has limited stretch, sheer and rotational components ($a_0 \approx a_3, a_1 \approx 0, a_2 \approx 0$). This assumption usually holds in reality; however, the instances where this assumption is violated, the progressive resolution improvement method can be used to improve performance. Furthermore progressive model improvement can be extended to the bilinear, parabolic or the perspective model to incorporate non-linear parameters into the algorithm.

6.4.4 Algorithm Description of PHGME

Having described the adaptation of the Hough transform for global motion estimation improvements to the original Hough transform, this section concludes with the description of the basic Hough transform algorithm. The following pseudo-code describes the basic 6-parameter HGME:

1. Divide the $\{a_0, a_1, a_2, a_3, a_4, a_5\}$ Hough space into 6-dimensional accumulator bins.
2. For each observation point $\{x, y, u, v\}$: do 3-4
3. For each a_0, a_1, a_2, a_3 quadruple: do 4
4. Given the current values of a_0, a_1, a_2, a_3, x, y, u, v, use Eq 6-10 to find a_4 and a_5. Increment the bin $[a_0][a_1][a_2][a_3][a_4][a_5]$.
5. Scan through all accumulator bins to locate the one with maximum value. The corresponding $\{a_0, a_1, a_2, a_3, a_4, a_5\}^*$ represents the quantized value of the global motion parameter set.

$$\begin{bmatrix} a_4 \\ a_5 \end{bmatrix} = \begin{bmatrix} u \\ v \end{bmatrix} - \begin{bmatrix} a_0 & a_1 \\ a_2 & a_3 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \qquad \text{Eq 6-10}$$

The 3-parameter HGME is similar, with the following differences:

Step 3 Iterating through a_0 only.

Step 4 Given the current values of a_0, x, y, u, v, use Eq 6-11 to find a_1, and a_2. Increment the bin $[a_0][a_1][a_2]$.

$$\begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} u \\ v \end{bmatrix} - \begin{bmatrix} a_0 & 0 \\ 0 & a_0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \qquad \text{Eq 6-11}$$

By incorporating the 3 improvements described in the previous sections into the basic Hough transform, the overall algorithm (referred to as progressive HGME, PHGME) can be described in Figure 6.11.

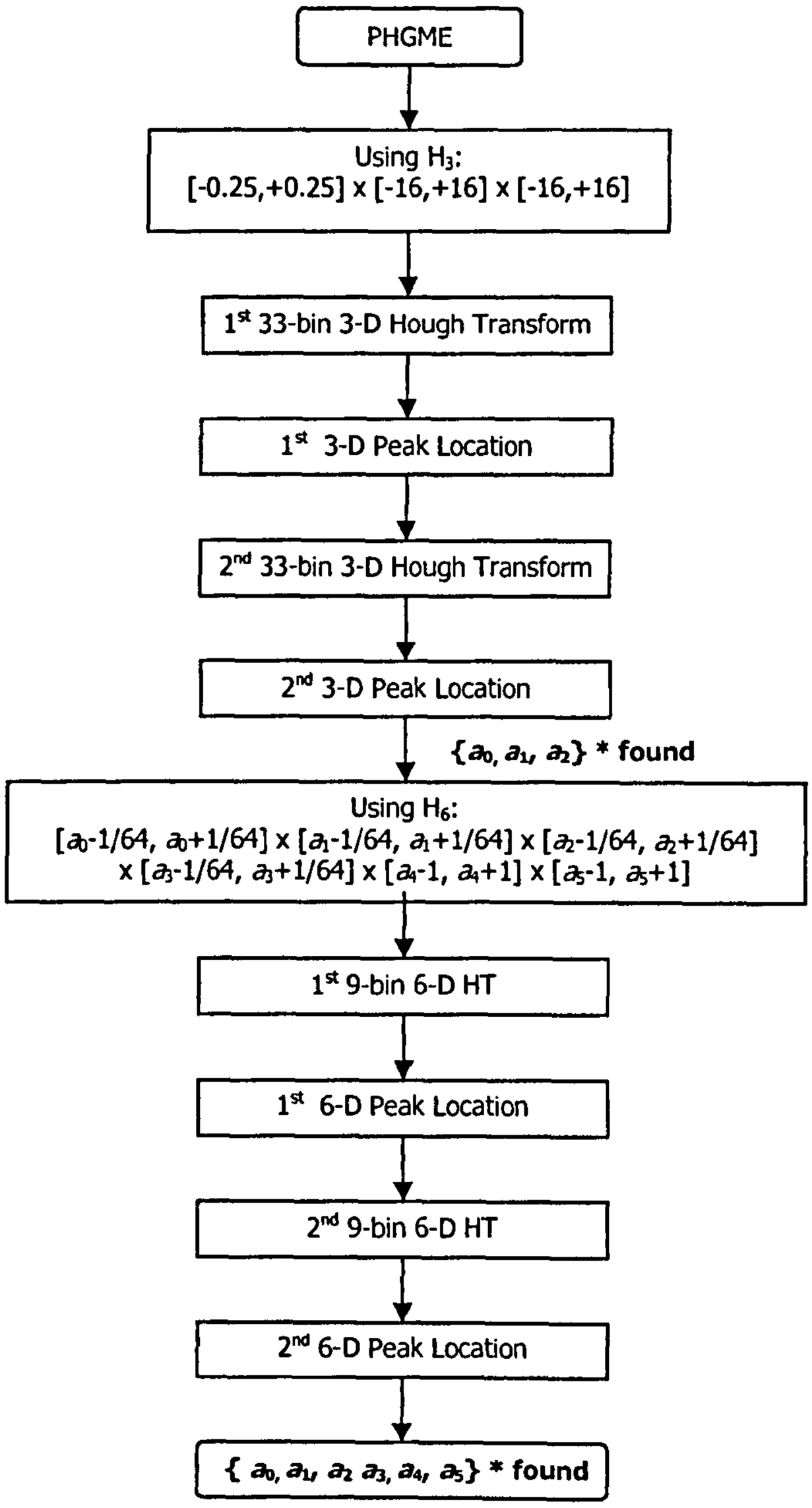


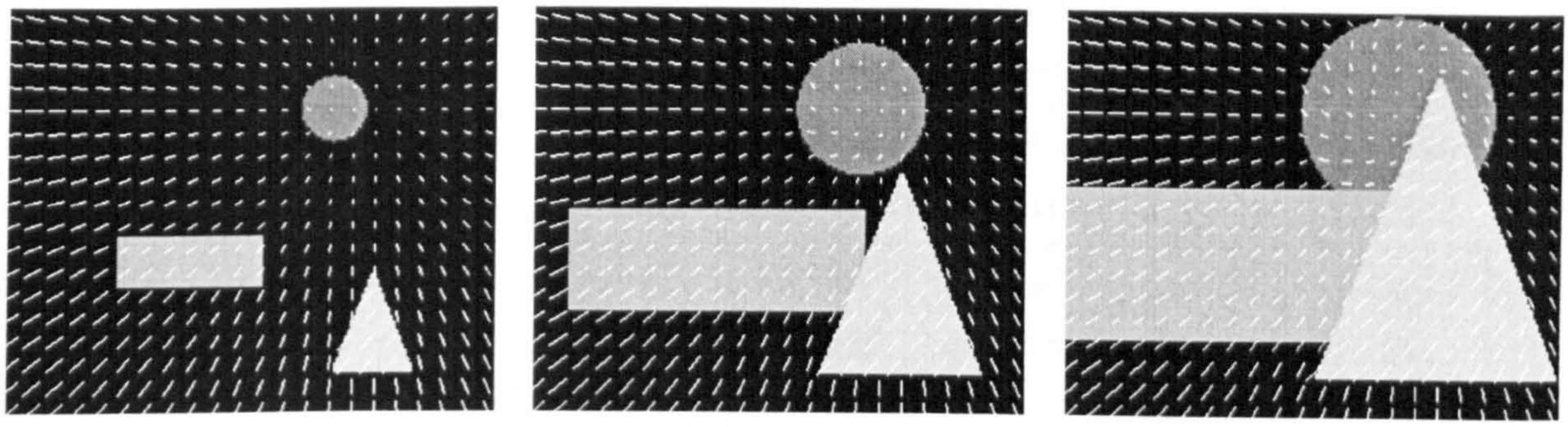
Figure 6.11. Flow chart of full PHGME algorithm.

6.5 Simulation Results

6.5.1 Synthetic Sequences

In order to compare performance the proposed the Progressive Hough transform (PHGME) is used to detect global motion parameters in presence of moving foreground objects, synthetic motion vectors of increasing larger moving objects are used.

With the synthetic fields, parameters of the global motion are compared with the 'ground truth' of the global motion present in the synthetic field. As a comparison, the results obtained from SIRGME described in the previous chapter are shown alongside those of PHGME. Three QCIF test frames with increasingly larger moving objects as shown in Figure 6.12 were used.



(a) F1 (b) F2 (c) F3

Figure 6.12. Test frame with background motion corrupted by locally moving objects.(a) F1: 92.49% background; (b) F2: 71.59% background; (c) F3: 49.31% background

The affine parameters of the objects and the background (ground truths) are shown in Table 6.1. Both SIRGME and PHGEM are used to estimate the background parameter. The motion vector is generated with 4×4 blocks.

Table 6.1 Motion models of objects and background.

Objects	a_0	a_1	a_2	a_3	a_4	a_5
Background	0.050	0.00	0.000	0.050	1.750	2.125
Rectangle	0.025	0.001	0.002	0.024	0.550	0.450
Circle	0.000	-0.065	0.065	0.000	2.250	1.833
Triangle	0.000	0.000	0.000	0.000	5.000	1.625

Table 6.2 shows the background parameters found by both SIRGME and PHGME methods. The deviation of the SIRGME parameters from the ground truth increases from F1 to F3, as the proportion of foreground objects increases. On the other hand, the PHGME parameters, deviates less from the ground truth.

Table 6.2 Motion models of objects and background.

Objects	a_0	a_1	a_2	a_3	a_4	a_5
Background	0.050	0.00	0.000	0.050	-1.750	2.125
SIRGME(F1)	0.04996	0.00000	0.00000	0.04999	-1.74998	2.12543
PHGME(F1)	0.05001	0.00000	0.00000	0.04989	-1.74107	2.12596
SIRGME(F2)	0.04930	-0.00019	0.00010	0.049866	-1.75592	2.12790
PHGME(F2)	0.04997	0.00000	0.00000	0.049949	-1.75378	2.13034
SIRGME(F3)	0.04731	-0.00055	0.00045	0.049672	-1.70290	2.13042
PHGME(F3)	0.05003	0.00000	0.00000	0.05003	-1.74107	2.12417

Table 6.3 summarizes the relative performances of PHGME with respect to SIRGME. The MSE deviation is the sum of square errors of the background parameters from ground truth. The scaling parameters (a_0, a_1, a_2, a_3) are multiplied by half the picture width (88 for QCIF) prior to the addition to the translational factors (a_4, a_5). From F1 to F3, the background proportions decrease, thus increasing the amount of noise for GME. The MSE of the SIRGME parameters increases progressively, whereas those of PHGME stay relatively constant. In F1, the background occupies 92.49% of the picture area. Both SIRGME and PHGME produce a global motion of very high accuracies, although the SIRGME parameters are slightly more accurate than those found with PHGME. This is mainly due to the fact the quantization effect of the PHGME, restricting the accuracy of PHGME. In F2 and F3, where the background areas are less dominant, the PHGME produces more accurate results, showing its resilience towards outliers.

Table 6.3 Accuracies of two GME algorithms to predict motion vectors in the background.

Test Field	Background proportion	MSE Deviation		Motion Field Entropy	
		SIRGME	PHGME	SIRGME	PHGME
F1	92.49%	3.64×10^{-7}	8.17×10^{-5}	0.9211	0.9011
F2	71.59%	9.27×10^{-5}	4.31×10^{-5}	3.3521	3.3242
F3	49.31%	2.94×10^{-3}	1.94×10^{-4}	6.3047	5.2188

In terms of a capability to compress the motion field, the entries under the 'motion field entropy' columns of Table 6.3 show the entropy of the remaining motion vector (to 1/4 –pixel accuracies) after removing the global motion components. All three test fields (F1, F2 and F3) carry less entropy under PHGME than SIRGME, showing the superiority of PHGME in terms of vector field compression.

6.5.2 Standard Sequences

Since natural test sequences lack ground truth, only the empirical method of comparing the relative performance between SIRGME and PHGME is adopted. PHGME and SIRGME are performed on six CIF and QCIF sequences and the results shown in Figure 6.13. All sequences produces less motion entropies with PHGME than with SIRGME. The two GME methods have relatively equal performances in TABLE sequences. This is due to the fact that (i) in the first part of the sequence the background constitutes a major part of the scene; (ii) the remaining part of the sequence is static. Both factors make both GME methods equally effective in producing an accurate parameter set. In general, PHGME brings about a bit rate reduction of about 0.05 bpp for QCIF@10fps and 0.02 bpp for CIF@30fps, or 12 kbps and 60 kbps savings respectively.

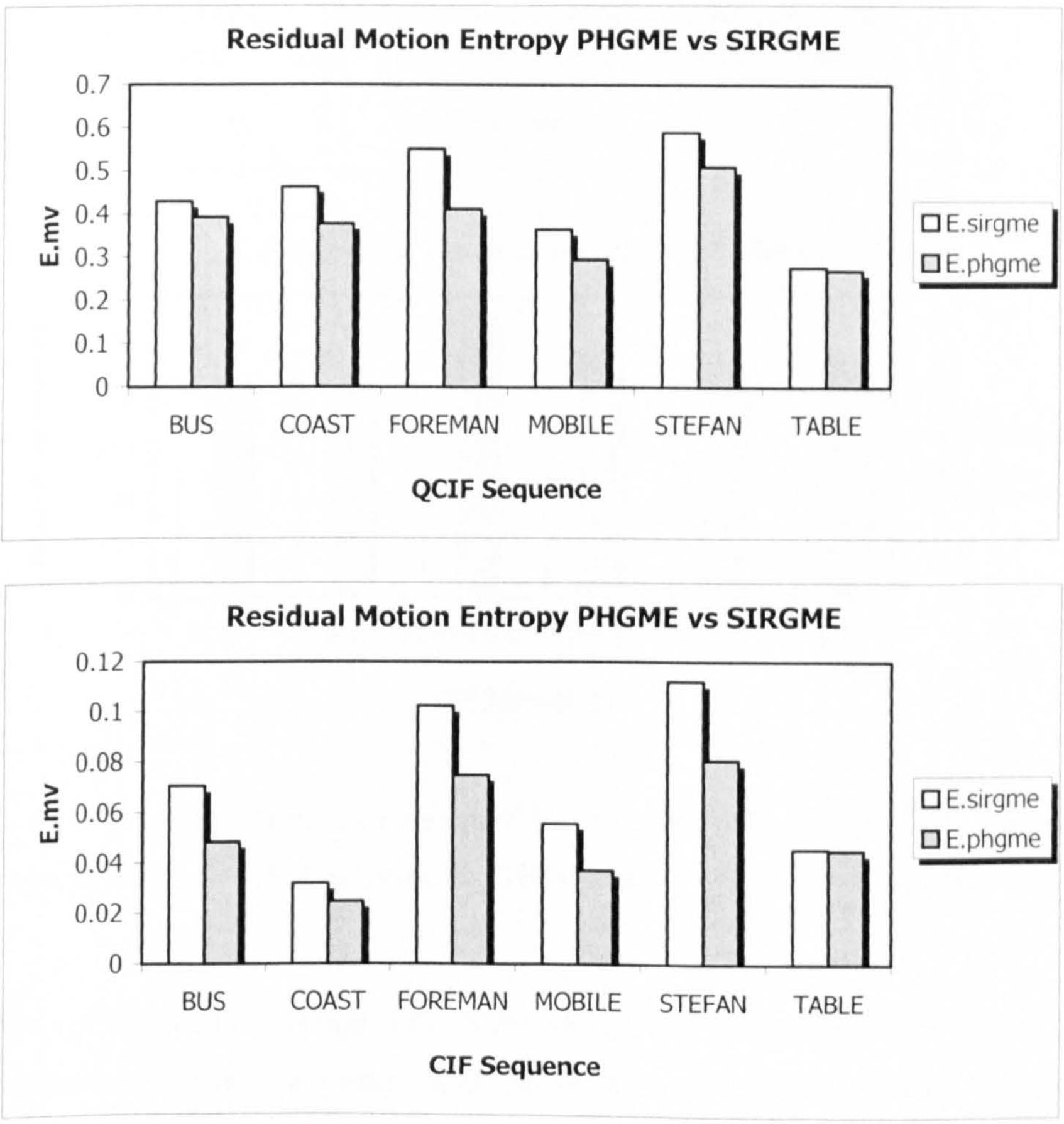


Figure 6.13. Motion entropies of six sequences from SIRGME and PHGME. Top: QCIF sequences; bottom: CIF sequences. PHGME outperforms SIRGME for all sequences.

Figure 6.14 shows that the processing times required by PHGME are about 0.4 seconds more than that of SIRGME. Hence, the robustness of PHGME is achieved at the expense of higher processing requirements. Although this additional time translates to a frame rate of a meagre 1 fps, the processing time is much lower than any traditional Hough Transform algorithms (usually beyond a minute per frame). Coupled with the low memory requirements, PHGME brings HT-based GME a large step closer to being adopted in real-time applications.

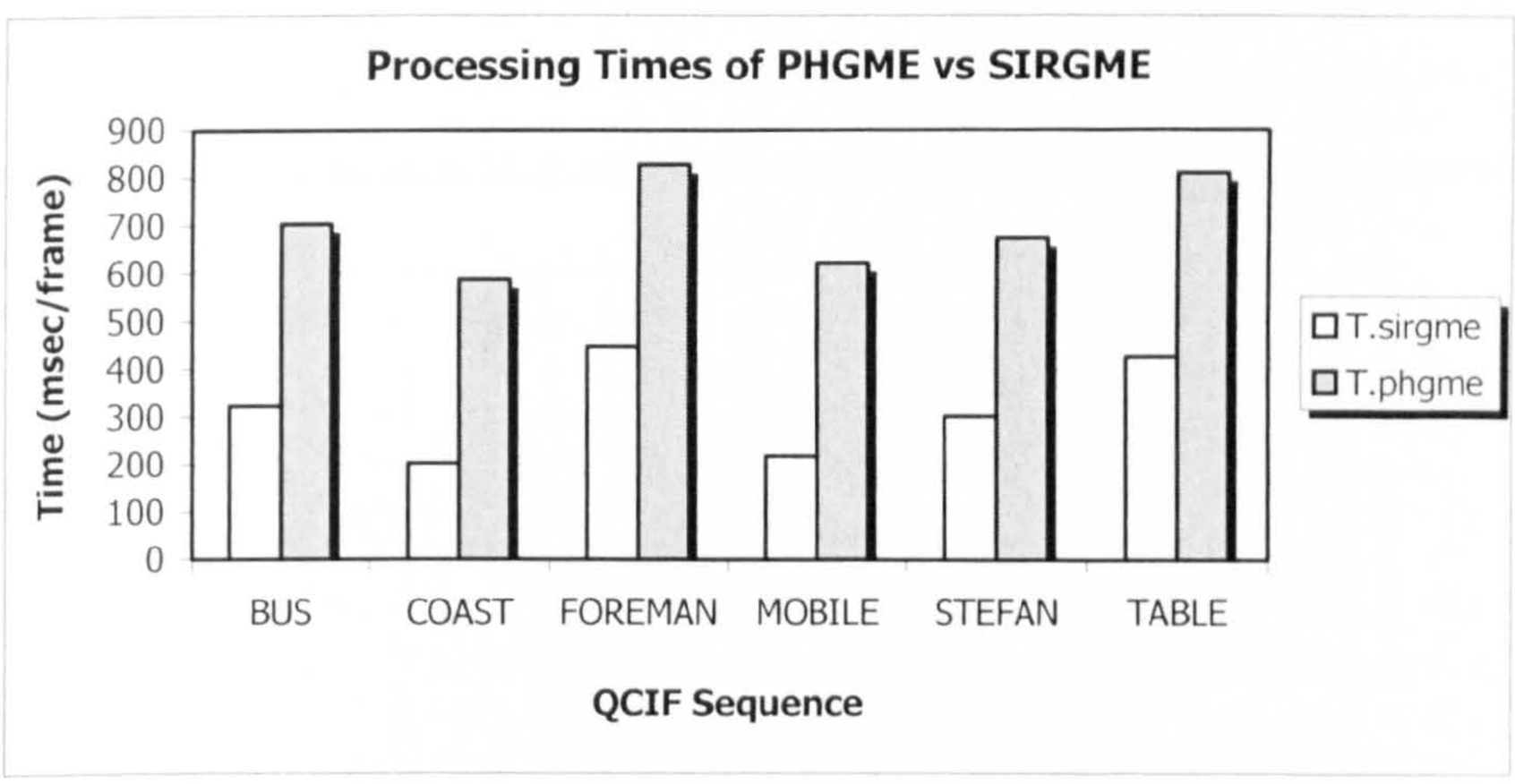


Figure 6.14. Processing times of six sequences from SIRGME and PHGME. Top: QCIF sequences; bottom: CIF sequences. PHGME takes longer time than SIRGME.

Lastly, the main operation of the Hough Transform, the polling of accumulators, can be made parallel where different parts of the motion vector field can be processed simultaneously. Compared with the iteration-based SIRGME, PHGME can be more easily adopted into DSP-based in hardware-based applications.

6.6 Conclusions

In this chapter, a novel global motion estimation method based on the Hough transform (HGME), the progressive Hough-based GME (PHGME), is presented. This method uses progressively lower resolution and more complex parameter models to achieve robustness towards outliers while keeping the processing and memory requirements low. Simulation results with both synthetic pictures and natural sequences show that the method produces a better estimate than the SAD-map-based Iterative Regressive GME (SIRGME) described in the previous chapter, especially in presence of relatively large moving objects. The algorithm's low memory requirements and processing speed, coupled with its parallel nature, make it suitable for real-time applications, both on DSP processors-based and hardware-based systems.

Chapter 7:

Motion Segmentation

Apparent motion in video sequences can be a result of the camera movements, or of independently moving objects. The previous two chapters dealt with video sequences with a dominant motion; for a sequence with several moving objects, the global motion estimation techniques may yield entirely meaningless results if the areas of foreground regions are significantly large. In such cases alternative approaches are required to segment more than two regions with uniform motions.

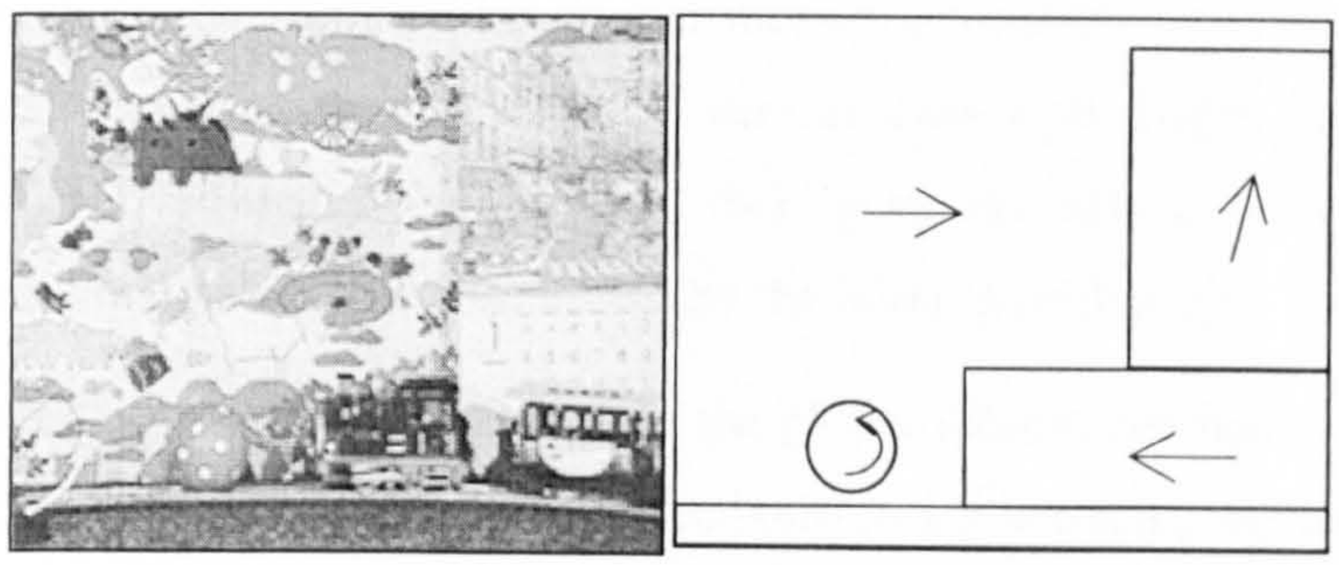


Figure 7.1. An illustration of motion segmentation using one frame from MOBILE sequence.

The simplest approach is used for the case where no camera motion can be assumed; all regional motions are caused by a few moving objects in a static background. In such case, moving regions are first extracted from a motion detection algorithm by frame differencing. The resulting "changed" pixels are then clustered into regions of uniform motion.

As motion segmentation (MotSeg) is an extension to global motion estimation for finding a set of dominant motion parameters, we can extend the global motion estimation algorithm by successively finding the dominant motion from the residual motion vector field with its previous dominant motion removed. Due to this natural extension, the same hardware and software components used in global motion estimation can be reused to solve the motion segmentation problem; the downside is its iterative nature makes parallel processing impossible. Another problem with this successive dominant motion estimation is the sensitivity of the algorithm to outliers. If the dominant motion in the previous step is influenced by another relatively large moving region, the inaccuracy of this result tends to propagates down to the next iteration step.

A more general approach to motion segmentation is by means of a statistical clustering on a set of motion vectors. This class of segmentation methods makes use of Bayesian statistics to maximize the a posteriori probability of a segmentation map given a motion vector field or the intensity fields of two pictures. Various methods vary according to the likelihood and the a-prior models of the segmentation map given the observed data.

The optimization of Bayesian statistics by annealing is usually computational untenable for real-time applications. A common alternative is to use the more deterministic method of clustering regions with similar model parameters. This results in either the K-means clustering or the Expectation-Maximization clustering. The latter method usually produces acceptable results with reasonable processing complexity.

Another class of motion segmentation uses the more heuristic split-and-merge approach to segmentation. This includes the variable block size motion estimation process which uses quad-tree decomposition by breaking big blocks into blocks of various sizes according to some motion uniformity criterion. The resulting quad-tree decomposition then goes through a merging process to fuse neighbours with similar motion which are separated by the initial partitioning.

Essentially, motion segmentation is an extension to the global motion estimation to more than one set of global motion parameters. In surveillance and recognition applications, motion segmentation is an indispensable precursor to the object recognition and object tracking steps. In video compression, motion segmentation reduces the amount of side information required to represent the motion field: by attributing motion in individual blocks to a handful of global motion parameters, only indices to a set of global motion parameters are required to represent the entire motion vector field.

In general, segmentation requires a similarity measure or distance measure between two regions. In image segmentation, the difference between the representative intensity levels, cluster centres or in the case of texture-based segmentation, weighted sum of the component differences can be used. In motion segmentation, how the motion of two regions resemble each other can be represented in terms of two distortion measures, as described in Eq 7-1. Both measures are based on the motion parameter θ_j of each region R_j. The first distortion measure, D_j (optical flow), is based on a measured optical flow or motion vector field, $\mathbf{v}(\mathbf{p})$, and the total difference of the vector compares to that due to the region's motion parameters, $\mathbf{v}_R(\mathbf{p}; \theta_j)$, denotes the region's distortion. The other measure, D_j(intensity), is based on the residual energy of the displaced frame difference (DFD) due to the motion region's parameters θ_j. Eq 7-1 describes the two measures:

$$D_j(\text{optical flow}) = \sum_{\mathbf{p}\in R_j} \left\| \mathbf{v}(\mathbf{p}) - \mathbf{v}_g(\mathbf{p};\boldsymbol{\theta}_j) \right\| \qquad \text{Eq 7-1}$$

$$D_j(\text{intensity}) = \sum_{\mathbf{p}\in R_j} \left\| I_t(\mathbf{p}) - I_{t-1}\left(\mathbf{p} - \mathbf{v}_g(\mathbf{p};\boldsymbol{\theta}_j)\right) \right\|$$

The next section provides an overview of various existing methods of motion segmentation based on these distortion measures.

7.1 Current Motion Segmentation Techniques

7.1.1 Foreground/Background segmentation

The most basic goal of motion segmentation is to separate the moving foreground from either a static background or an apparently moving background due to camera motion [Mos-95] [Zha-97] [Jin-00] [Ben-94].

The simplest form of motion segmentation uses motion detection, for example, to determine if there is non-zero global motion. A simple frame-differencing step extracts moving regions based on residual energy. An adaptive threshold is then implemented. Several methods for determining optimal thresholding have been published [Kim-99b] [Dur-00] [Hua-02]. The changed regions are typically scattered in space and various merging methods have been proposed to group these isolated changed points into contiguous moving regions. Kim and Hwang [Kim-99b] propose a novel means of detecting changed regions with three edge maps using the Canny edge detection algorithm. They subsequently produce moving Video Object Planes (VOP, an acronym used in MPEG-4 object-based coding) using a logical-OR operation on horizontal VOP candidates and vertical VOP candidates, which are points inside the first and last edge points for each row and column respectively. Erasing mask resembling the morphological opening operation is used to remove small regions. The authors provide simulation results with the HALL sequence which shows the successful extraction of the two moving bodies.

When a non-zero background motion is assumed, a global motion estimation step is carried out to identify the dominant motion in each frame. Foreground regions are then identified by either a large deviation of the local motion from that due to the estimation global motion [Mos-95], or a large mismatch in intensity of the region and the region from the referenced frame warped according to the global motion model [Jin-00].

Foreground/background segmentation is essentially a thresholding problem. In applications with static camera [Kim-99b] [Hua-02], change detection techniques are used. The former used edge constancy measure and the latter used intensity constancy measure. In the case of sequences with global motion [Duf-95b] [Zha-95], the reference picture is firstly warped using the global motion parameters. The

warped reference frame is a better match with the input frame; subsequently motion detection can be performed between the input frame and the warped reference frame. Zhang [Zha-95] further improved the performance by making the threshold value adaptive. Jinzenji et. al. [Jin-00] aligned several warped frames and used rank statistic to determine the change mask. Once foreground regions are extracted, the can be further segmented through any methods described below.

It is noted that in the process of regressive estimation of the global motion, outliers are usually identified and their influence removed from further iterations. Hence the outliers in each round of iterative represent a likely set of foreground regions. The confidence level of the foreground would be increased if we included the intersection of outliers of all iterations.

7.1.2 Successive Dominant Motion Elimination

The foreground/background extraction can be viewed as a process to extract the dominant motion from a mixture of smaller ones. This approach can be extended by first removing the region of dominant motion, and performing the dominant motion iteratively with the remaining regions until all regions are removed or the residual region cannot be attributed to a single motion. Such a process is termed successive dominant motion elimination, first used by Irani and Peleg in [Ira-92].

A typical implementation is proposed by Borshukov et. al. [Bor-97], where a dense optical flow is obtained and then divided into non-overlapping rectangular blocks, and each block is assigned a 6-parameter affine model via regression. The block with the smallest regression error is selected and other blocks having similar parameter values are marked; unmarked clusters then undergo a similar process until all blocks are marked. *K*-means clustering is then used to refine the segmentation and parameter values.

In [Pel-90], dominant motion estimation is found and denoted as θ_0 and a warped reference is subtracted from the input. Regions with high residues are grouped and the largest of such region goes through another round of global motion estimation resulting θ_1. A novel ratio is for each pixel **p** defined as:

$$R(\mathbf{p}) = \frac{\left| I_t(\mathbf{p}) - I_{t-1}\left(\mathbf{p} - \mathbf{v}_g(\mathbf{p};\theta_1)\right)\right|}{\left| I_t(\mathbf{p}) - I_{t-1}\left(\mathbf{p} - \mathbf{v}_g(\mathbf{p};\theta_0)\right)\right|} \qquad \text{Eq 7-2}$$

A pixel **p** will be classified as background if $R(\mathbf{p}) >> 1$; **p** is classified as foreground if $R(\mathbf{p}) << 1$ and **p** is unclassified when $R(\mathbf{p}) \approx 1$. After removing classified points, θ_1 is used as an initial estimate of the next iteration to find θ_0 and θ_1. By successively performing this procedure, progressively smaller moving objects are extracted.

7.1.3 Clustering with Motion Similarity Measure

In contrast to previous methods, where the whole image is broken down into segments in a top-down manner, the following methods use the bottom-up method where small regions are merged according to a certain similarity in the regions' motion parameters. The similarity measures are based on how the distortion in Eq 7-1 changes before and after merging. A decrease in D_{ij} from D_i+D_j implies a good similarity between R_i and R_j. The typical method of region merging as outlined as:

- Specify initial M segments.
- Specify a target number of clusters, *K*.
- Start off with M clusters; merge two adjacent clusters C_i and C_j with the largest similarity.
- Repeat previous step until number of clusters is reduced to K.
- Use linear regression or other iterative methods to update the *K* cluster properties.

A typical case is proposed by [Ade-94], in which a dense motion vector field is clustered via the *k*-means clustering algorithm. In their proposed method, the optical field is initially divided into square blocks and affine parameters within each block are estimated by the standard least-mean-squares linear regression technique. For each set of parameters, its regression residual error is compared with a prescribed threshold. Parameter sets whose residual errors are above the threshold are regarded as incorrect and discarded. The remaining parameter sets form the initial cluster centres for the *k*-means clustering algorithm. Adelson and Wang pointed out that the Euclidean distance of the affine-parameter set is not suitable for the clustering algorithm and modified the distance measure by scaling the four scale/rotation parameters (a_0, a_1, a_2 and a_3 of Eq 5-23) so that the a unit distance along any component in the parameter space corresponds to roughly a unit displacement at the picture boundaries. By using this distance measure, pixels are assigned to the cluster whose parameters produce a motion vector closest to the optical flow value. Cluster parameters are recalculated by linear regression of the new clusters and the assignment-regression process is iterated.

During some steps of the iterative process, some centres may converge. When the distance between any two centres is less than a threshold, the two clusters are merged into a single centre, thus reducing the number of clusters. In this way the number of clusters is made adaptive where a small number of clusters are maintained while keeping the distortion small.

Robustness of the algorithm is improved by identifying pixels whose motion cannot be adequately described by any cluster as outliers. The outliers are removed from future steps. The iteration stops when either less than a certain number of pixels are re-assigned, or the maximum iteration count is reached. Outlier pixels are then re-assigned by warping the images according to the global motion model that minimizes the intensity error between the input and reference images. In [Ngu-00], Nguen, Worring and Dev put emphasis on the definition of a new motion similarity measure and a novel

merging process through hypothesis testing. In contrast to [Ade-94] where a fixed scale is used in the distance measure, they proposed a similarity that took the statistical uncertainty of the respective parameters into account, which they claimed to be invariant to a change in the origin of the image grid. In their paper, the quadratic model is used and model parameters $\boldsymbol{\theta}_i = \{a_0, a_1, \cdots, a_7\}_i$ of each region R_i are found by direct methods using optical flow equation:

$$\hat{\boldsymbol{\theta}}_i = \arg\min_{\theta} S_i(\boldsymbol{\theta}) \quad \text{where} \quad S_i(\boldsymbol{\theta}) = \sum_{\mathbf{p}\in R_i} [e(\mathbf{p};\boldsymbol{\theta})]^2 \qquad \text{Eq 7-3}$$

$$e(\mathbf{p};\boldsymbol{\theta}) = \frac{\partial I_t(p)}{\partial t} + \frac{\partial I_t(p)}{\partial x} v_x + \frac{\partial I_t(p)}{\partial y} v_y$$

$$= y(\mathbf{p}) - \begin{bmatrix} I_x & xI_x & yI_x & I_y & xI_y & yI_y & xyI_x + y^2 I_y & x^2 I_x + xyI_y \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_7 \end{bmatrix}$$

An initial region set is obtained by *k*-means clustering based on colour/intensity similarity. Based on the assumption that motion edges form a subset of colour/intensity edges, this region set forms the initial segmentation for the merging algorithm.

Nguen, et. al. claimed that distance measures with constant scales like the one proposed in [Ade-94] have a few short-comings and proposed a measure that resembled the Mahalanobis distance. Initially, least-squares minimization is used to estimate the motion parameter of each input region, just to find out the variance of the noise robustly using Eq 7-4 where $r_{LS}(x)$ is the least-squares residual and *med* is the median function:

$$\hat{\sigma} = 1.48 med |r_{LS}(x)| \qquad \text{Eq 7-4}$$

The standard deviation $\hat{\sigma}$ is then used subsequently in each iteration step where the Huber's M-estimator is used in place of Eq 7-3:

$$\hat{\boldsymbol{\theta}}_i = \arg\min_{\theta} \sum_{\mathbf{p}\in R_i} \rho\left(\frac{I_t(\mathbf{p}) - I_{t-1}(\mathbf{p};\boldsymbol{\theta})}{\hat{\sigma}}\right) \qquad \text{Eq 7-5}$$

$$\rho(e) = \begin{cases} e^2 & \text{if } |e| < c \\ c(|e| - c/2) & \text{otherwise} \end{cases}$$

The core of their merging algorithm is based on representing the likelihood of a region by a normal distribution of the DFD, and on using a hypothesis test of whether the likelihood of two regions are lowered when they are merged:

$$H_0: \theta_i = \theta_j; \quad H_1: \theta_i \neq \theta_j \qquad \text{Eq 7-6}$$

The symbols θ_i and θ_j represent the motion model parameters of region i and j respectively. The test statistic used is the likelihood ratio:

$$\Delta_{ij} = -2\log\left(\frac{\sup_{\theta_i=\theta_j} L}{\sup_{\theta_i\neq\theta_j} L}\right) \qquad \text{Eq 7-7}$$

$$= \frac{1}{\hat{\sigma}^2}\left\{\min_{\theta}\left[S_i(\theta)+S_j(\theta)\right] - \min_{\theta} S_i(\theta) - \min_{\theta} S_j(\theta)\right\}$$

$$= \frac{1}{\hat{\sigma}^2}\left[\hat{S}_{ij} + \hat{S}_i + \hat{S}_j\right]$$

Two regions R_i and R_j are merged if Δ_{ij} is below a threshold T. The significant contribution by the paper is the realization that the solution to Eq 7-3 is:

$$\mathbf{y} = \mathbf{f}\theta \;\Rightarrow\; \hat{\theta} = \left(\mathbf{f}^T\mathbf{f}\right)^{-1}\mathbf{f}^T\mathbf{y} \qquad \text{Eq 7-8}$$

$$\mathbf{y} = \begin{bmatrix} -\dfrac{\partial I(\mathbf{p})}{\partial t}\Big|_{\mathbf{p}=\mathbf{p}_1} \\ \vdots \end{bmatrix}$$

$$\mathbf{f} = \begin{bmatrix} I_x(\mathbf{p}_1) & x_1 I_x(\mathbf{p}_1) & y_1 I_x(\mathbf{p}_1) & I_y(\mathbf{p}_1) & x_1 I_y(\mathbf{p}_1) & y_1 I_y(\mathbf{p}_1) & x_1 y_1 I_x(\mathbf{p}_1) + y_1^2 I_y(\mathbf{p}_1) & x_1^2 I_x(\mathbf{p}_1) + x_1 y_1 I_y(\mathbf{p}_1) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

For each region pair R_i, the Hessian is defined as $\mathbf{H}_i = 2\mathbf{F}_i^T\mathbf{F}_i$. The authors noted that by using the Hessians, the merged parameters, the errors and the new Hessians are linked with corresponding parameters of the regions prior to merging:

$$\hat{S}_{ij} = \hat{S}_i + \hat{S}_j + \frac{1}{2}\left(\hat{\theta}_i - \hat{\theta}_j\right)^T \mathbf{H}_i\left(\mathbf{H}_i + \mathbf{H}_j\right)^{-1}\mathbf{H}_j\left(\hat{\theta}_i - \hat{\theta}_j\right) \qquad \text{Eq 7-9}$$

$$\hat{\theta}_{ij} = \left(\mathbf{H}_i + \mathbf{H}_j\right)^{-1}\left(\mathbf{H}_i\hat{\theta}_i + \mathbf{H}_j\hat{\theta}_j\right)$$

$$\mathbf{H}_{ij} = \mathbf{H}_i + \mathbf{H}_j$$

$$\Delta_{ij} = \frac{1}{2\hat{\sigma}^2}\left(\hat{\theta}_i - \hat{\theta}_j\right)^T\left(\mathbf{H}_i^{-1} + \mathbf{H}_j^{-1}\right)^{-1}\left(\hat{\theta}_i - \hat{\theta}_j\right)$$

By using Eq 7-9, merging can be done rapidly. When all region pairs whose H_1 are rejected are merged, the remaining clusters undergo the standard k-means clustering algorithm to obtain the final segmentation and parameters.

Prior to merging, Liu and Hayes [Liu-92] added a splitting process, arriving at a split-and-merge algorithm. The split procedure employed a quad-tree hierarchical data structure using a predefined uniformity criterion to decide if a region is to be split. The merging process is based on the output of the previous procedure. By the help of weighted region adjacency graph (WRAG), split regions can be merged across entries in the root tree by means of motion similarity defined as the distortion change before and after merging. Other similar method using RAG include [Stu-92], [Sch-96] and [Li-01].

It is also very common to incorporate colour information into the whole merging process to produce a joint motion/texture segmentation process. This is typified by Altunbasak et. al's paper [Alt-98].

7.1.4 Hough Transform –based Motion Segmentation

The Hough transform was used in the previous chapter to detect the dominant motion. It can be generalized as a motion segmentation tool by detecting multiple peaks in the Hough space. Adiv [Adi-85] first used the technique on a 6-dimensional space to detect moving regions with affine model. To reduce computations, the 6-D Hough space is split into two 3-D Hough space and each motion vector votes only once to each Hough space. A multi-resolution approach similar to that used in the previous chapter was adopted. A multi-pass approach is used where the transform is performed many times, each time eliminating votes from previously detected values. By setting a threshold for detecting motion, Adiv's algorithm can determine the number of segments automatically. An alternative is provided by Bober and Kittler [Bob-93] whereby the direct method is used in place of the indirect method based on optical flow.

There is a separate approach to Hough-based motion segmentation in which edge and line features are matched between two frames. The correspondences between these edges can be used to poll for translation, zoom and rotation parameters. Different versions of Hough transforms are used in various papers. These include the adaptive Hough transform [Tia-95], the generalized Hough transform [Sil-98] and the randomised Hough transform [Kal-96]. One common step throughout different variants of this approach is the selection of good features points for matching and all of them are extensions of edge finding algorithms and is more relevant in machine recognition of good shaped objects in motion; their use in video coding systems is limited.

7.1.5 Motion Segmentation by Bayesian Methods

The Bayesian methods of segmentation search for the most probable labelling configuration given the motion vector field, which measures how well the current segmentation labelling explains the vector

field and how well the field conforms to some prior expectations. We define a label field $\mathbf{z}(\mathbf{p}) \in \{0,1,\ldots,J-1\}$ where J is the number of segments. In its simplest form, given a motion vector field $\mathbf{v}(\mathbf{p})$, the a posterior probability of a segmentation $\mathbf{Z}$ where $\mathbf{Z} \in \{0,1,\ldots,J-1\}^K$, K is the number of regions, given a motion vector field $\mathbf{V}$ can be expressed in terms of the likelihood and the prior probability:

$$p(\mathbf{Z}\,|\,\mathbf{V},\mathbf{I}_t,\mathbf{I}_{t-1}) = \frac{p(\mathbf{V}\,|\,\mathbf{Z},\mathbf{I}_t,\mathbf{I}_{t-1})p(\mathbf{Z}\,|\,\mathbf{I}_t,\mathbf{I}_{t-1})}{p(\mathbf{V}\,|\,\mathbf{I}_t,\mathbf{I}_{t-1})} \qquad \text{Eq 7-10}$$

where $\mathbf{I}_t$ and $\mathbf{I}_{t-1}$ are the intensity fields of the current and previous frames respectively. This method is a natural extension of the Bayesian methods for global motion estimation in which the prior and likelihood probabilities are modelled analytically given extra constraints. Various simulated and deterministic annealing methods can be used to optimize the a posteriori probabilities.

The method is typified by [Mur-87], where Murray and Buxton used simulated annealing method to obtain $\mathbf{Z}$ with M quadratic models based on a dense optical flow $\mathbf{V}$. In their paper, the likelihood probability is modelled according to the difference between the optical flow and the displacement induced by the motion parameter set, which is assumed to be Gaussian distributed with zero mean and variance σ:

$$p(\mathbf{V}\,|\,\mathbf{Z},\mathbf{I}_t,\mathbf{I}_{t-1}) = \frac{1}{\left(2\pi\sigma^2\right)^{M/2}} \exp\left[-\sum_{i=1}^{M}\sum_{p\in R_i}\left\|v_{op}(\mathbf{p}) - v_{gm}(\mathbf{p};\theta_i)\right\|^2\right] \qquad \text{Eq 7-11}$$

On the other hand, the prior probability is modelled as a Gibbs distribution of clique potentials.

Alternative to Eq 7-10, both $\mathbf{Z}$ and $\mathbf{V}$ can be simulated by direct methods with:

$$p(\mathbf{Z},\mathbf{V}\,|\,\mathbf{I}_t,\mathbf{I}_{t-1}) = \frac{p(\mathbf{I}_t\,|\,\mathbf{V},\mathbf{Z},\mathbf{I}_{t-1})p(\mathbf{V}\,|\,\mathbf{Z},\mathbf{I}_{t-1})p(\mathbf{Z}\,|\,\mathbf{I}_{t-1})}{p(\mathbf{I}_t\,|\,\mathbf{I}_{t-1})} \qquad \text{Eq 7-12}$$

This method is usually termed simultaneous motion estimation and segmentation [Cha-97], where the optical field, the segmentation and motion parameters are found simultaneously through MAP (maximum a-posteriori probability) estimation. The first conditional probability in Eq 7-12 constrains the motion vector field $\mathbf{V}$ be minimizing DFD. The second conditional probability constrains the motion parameters by minimizing the difference between the vectors in $\mathbf{V}$ and the vector field obtained by current segmentation $\mathbf{Z}$ and the motion parameters of the regions. The last term in the numerator

gives more priority to more uniform segmentation. The process is usually initialized by obtaining an optical flow and some segmentation of a regular grid.

Chang et al [Cha-97] has used Eq 7-12 in an effective manner and combined with deterministic annealing methods (HCF and ICM) and obtained good results using a reasonable amount processing resources. In their paper, the three likelihood and prior terms are structured as exponential distributions which can be simplified to:

$$-E(\mathbf{V},\mathbf{Z}\,|\,\mathbf{I}_t,\mathbf{I}_{t-1}) = U_1(\mathbf{I}_t\,|\,\mathbf{V},\mathbf{Z},\mathbf{I}_{t-1}) + U_2(\mathbf{V}\,|\,\mathbf{Z},\mathbf{I}_{t-1}) + U_3(\mathbf{Z}\,|\,\mathbf{I}_{t-1}) \qquad \text{Eq 7-13}$$

$$U_1(\mathbf{I}_t\,|\,\mathbf{V},\mathbf{Z},\mathbf{I}_{t-1}) = \sum_{all\,\mathbf{p}} \left[I_t(\mathbf{p}) - I_{t-1}(\mathbf{p};\boldsymbol{\theta}_{z(\mathbf{p})})\right]^2$$

$$U_2(\mathbf{V}\,|\,\mathbf{Z},\mathbf{I}_{t-1}) = \alpha \sum_{all\,\mathbf{p}} \left[\mathbf{v}(\mathbf{p}) - \mathbf{v}_{gm}(\mathbf{p};\boldsymbol{\theta}_{z(\mathbf{p})})\right]^2 + \beta \sum_{all\,\mathbf{p}} \sum_{\mathbf{q}\in N(\mathbf{p})} \left[\mathbf{v}(\mathbf{p}) - \mathbf{v}(\mathbf{q})\right]^2 \delta\big(z(\mathbf{p}) - z(\mathbf{q})\big)$$

$$U_3(\mathbf{Z}\,|\,\mathbf{I}_{t-1}) = \gamma \sum_{all\,\mathbf{p}} \sum_{\mathbf{q}\in N(\mathbf{p})} \left[1 - 2\delta\big(z(\mathbf{p}) - z(\mathbf{q})\big)\right]$$

In Eq 7-13, U_1 is the DFD of the current pixel and the pixel in previous frame displaced by the segment's motion parameters. U_2 has two terms – the first term binds the current estimate of **V** and that of the segmentation and its accompanying parameter estimates; the second term implements the smoothness constraint of **V** within the segment. The Kronecker function removes the influence of motion boundary pixels. U_3 uses potentials of two-member cliques of segmentation estimates to encourage contiguous regions. In order minimize E, [Cha-97] uses an iterative approach which, alternates the estimation of **V** and **Z**, both using deterministic annealing (HCF).

Several other motion analysis algorithms are formulated as special cases of Eq 7-12 and Eq 7-13. Removing the second term of Eq 7-12 converts the problem into the classic Bayesian-based local motion estimation. Iu [Iu-93] used the same remaining two terms, but replaced the Kronecker function by outlier rejection. Stiller [Sti-94] did similar omission with the aim to just provide piecewise smoothness constraint of the optical flow. Murray and Buxton [Mur-87] used the α and γ terms in Eq 7-13 to model the likelihood and prior probabilities respectively. Vasconcelos and Lippman [Vas-01] used U_1 and U_3 with higher clique potentials and incorporate them into a empirical Bayesian framework.

An interesting alternative to using the segmentation map **Z**, Heitz and Bouthemy [Hei-90], and J. Konrad and E. Dubois [Kon-92] used edges to partition regions of interest. Consequently edge likelihoods are used instead of segment smoothness constraints. It was claimed that the algorithm produces a high resemblance to the ground trut, especially near motion discontinuities.

Undoubtedly, simultaneously obtaining regions and vector fields in general yields better results than other methods; a major drawback with this method is that optimization becomes excessively more complicated than the already complex form of Eq 7-10. A compromise is the expectation-maximization (EM) algorithm which will be discussed in the following sections. The necessary theories will be laid out subsequently before the novel EM-based motion segmentation algorithm is introduced. Simulation results from both synthetic and test sequences will be analyzed.

7.2 Motion Segmentation by Expectation-Maximization

7.2.1 Basics of Expectation-Maximization

Expectation-Maximization [Lai-77] is a well-known numerical algorithm for maximizing functions of several variables. It is commonly used to obtain a maximum-likelihood estimate of a set of parameters for an underlying distribution from which a given set of data is incomplete or partially observable.

In the maximum likelihood estimation problem, let $\mathbf{Z}$ be an random vector whose distribution is conditional to the set of parameters Θ whose value we need to estimated given an observed instance of $\mathbf{Z}$. Maximum-likelihood estimation states that the optimal estimator Θ^* is one that produces the maximum likelihood $L(\Theta|\mathbf{Z})$:

$$L(\Theta|\mathbf{Z}) = p(\mathbf{Z}|\Theta) \qquad \text{Eq 7-14}$$
$$\Theta^* = \arg\max_{\Theta} L(\Theta|\mathbf{Z}) = \arg\max_{\Theta} p(\mathbf{Z}|\Theta)$$

Now assume that $\mathbf{Z}$ is not entirely observable, and is made up of two components $\mathbf{X}$ and $\mathbf{Y}$. $\mathbf{X}$ is the part of data which is observable and $\mathbf{Y}$ is the other part which is either unobservable or too complex to be measured directly. The former is called the observed data and the latter is called the hidden data. Together, $\mathbf{X}$ and $\mathbf{Y}$ form the complete data such that $p(\mathbf{Z}) = p(\mathbf{X},\mathbf{Y})$. Now the complete-data likelihood becomes:

$$L(\Theta|\mathbf{X},\mathbf{Y}) = p(\mathbf{X},\mathbf{Y}|\Theta) = p(\mathbf{Z}|\Theta) \qquad \text{Eq 7-15}$$

Note that $L(\Theta|\mathbf{X},\mathbf{Y})$ is in fact a probability function since the missing information $\mathbf{Y}$ is unknown, and hence can be modelled as a random variable. So we can think of $L(\Theta|\mathbf{X},\mathbf{Y})$ as a function of $\mathbf{Y}$ and $\mathbf{X}$ (observed data) whilst Θ (the parameter set whose value we want to determine) are fixed. The maximium likehood problem can be changed to maximize the expected value of the likehood, hence the name Expectation-Maximization.

The EM algorithm is an iterative process involving two steps. The first step finds the expected value of the complete-data log-likelihood $\log[p(\mathbf{X},\mathbf{Y}|\Theta)]$ with respect to the missing data $\mathbf{Y}$ given the observed data $\mathbf{X}$ and the current parameter estimate, $\hat{\Theta}$, defined as $Q(\Theta|\hat{\Theta})$

$$Q(\Theta|\hat{\Theta}) = E_{\mathbf{Y}|\mathbf{X},\hat{\Theta}}[\log p(\mathbf{X},\mathbf{Y}|\Theta)] \qquad \text{Eq 7-16}$$

If $\mathbf{Y}$ is discrete, $Q(\Theta|\hat{\Theta})$ can be expressed as:

$$Q(\Theta|\hat{\Theta}) = \sum_{\text{all } \mathbf{Y}} \log(p(\mathbf{X},\mathbf{Y}|\Theta))p(\mathbf{Y}|\mathbf{X},\hat{\Theta}) \qquad \text{Eq 7-17}$$

Note that $p(\mathbf{Y}|\mathbf{X},\hat{\Theta})$ is the marginal distribution of the unobserved data and is dependent on both the observed data $\mathbf{X}$ and on the current parameter estimate. The evaluation of this expectation as a function of Θ is called the E-step in the EM-algorithm. The second step, the M-step, is to maximize this expectation with respect to the parameter space $\{\Theta\}$:

$$\Theta^* = \arg\max_{\text{all } \Theta} Q(\Theta|\hat{\Theta}) \qquad \text{Eq 7-18}$$

The new estimated parameter set Θ^* is then used to perform the next E-step. These two steps are usually repeated until Θ^* converges within a given tolerance or a maximum number of steps is reached:

$$Q(\Theta|\Theta^{n-1}) = \sum_{\text{all } \mathbf{Y}} \log p(\mathbf{X},\mathbf{Y}|\Theta)p(\mathbf{Y}|\mathbf{X},\Theta^{n-1}) \qquad \text{Eq 7-19}$$

$$\Theta^n = \arg\max_{\text{all } \Theta} Q(\Theta|\Theta^{n-1})$$

Theoretically, each iterative step is guaranteed to increase the log-likelihood and the algorithm is guaranteed to converge to a local maximum of the likelihood function. A modified form of the M-Step is to, instead of maximize $Q(\Theta|\Theta^{n-1})$, find some Θ^n such that $Q(\Theta^n|\Theta^{n-1}) > Q(\Theta^{n-1}|\Theta^{n-2})$. This form of algorithm is termed Generalized EM (GEM) and is also guaranteed to converge, albeit at a lower rate.

Out of various EM-related problems, the mixture-density parameter estimation and the hidden Markov model are two commonest. We shall focus on the former in the remaining part of this chapter; in the next section, we shall describe how it is can be adapted in performing motion segmentation using a sparse motion vector field as the observable data set.

In a mixture-model, the probability of a random variable **X** is composed of *J* distributions:

$$p(\mathbf{X}|\Theta) = \sum_{j=1}^{J} \alpha_j p_j(\mathbf{X}|\theta_j) \qquad \text{Eq 7-20}$$

$$\sum_{j=1}^{J} \alpha_j = 1$$

In Eq 7-20, each $p_j(\mathbf{X}|\theta_j)$ is a probability density function parameterized by θ_j; α_j is the mixing proportion. Usually **X** is a random vector of *K* independent and identically distributed (iid) random variables: $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_K\}$

$$p(\mathbf{X}|\Theta) = \prod_{k=1}^{K} p(\mathbf{x}_k|\Theta) \qquad \text{Eq 7-21}$$

$$= \prod_{k=1}^{K} \sum_{j=1}^{J} \alpha_j p_j(\mathbf{x}_k|\theta_j)$$

And the incomplete-data log-likelihood can be expressed as:

$$\log(L(\Theta|\mathbf{X})) = \log\left[\prod_{k=1}^{K} p(\mathbf{x}_k|\Theta)\right] \qquad \text{Eq 7-22}$$

$$= \sum_{k=1}^{K} \log \sum_{j=1}^{J} \alpha_j p_j(\mathbf{x}_k|\theta_j)$$

Further assume that the mixtures are Gaussian distributions, where $\theta_j = (\mathbf{m}_j, \Sigma_j)$, $\mathbf{m}_j$ and Σ_j being the component's mean vector and covariance matrix respectively. The conditional probability of each mixture component $p_j(x_k|\theta_j)$ becomes:

$$p_j(\mathbf{x}_k|\theta_j) = p_j(x_k|\mathbf{m}_j, \Sigma_j) = \frac{1}{\sqrt{(2\pi)^D|\Sigma_j|}} \exp\left[-\frac{1}{2}\mathbf{m}_j{}^T \Sigma_j{}^{-1} \mathbf{m}_j\right] \qquad \text{Eq 7-23}$$

Where *D* is $\mathbf{m}_j$ and $|\Sigma_j|$ is the matrix determinant of Σ_j. It is not difficult to see that the hidden data set is: $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_J\}$ where $\mathbf{y}_j = \alpha_j$. The complete-data log-likelihood then becomes:

$$\log(L(\Theta|\mathbf{X},\mathbf{Y})) = \sum_{k=1}^{K} \log \sum_{j=1}^{J} \frac{\alpha_j}{\sqrt{(2\pi)^D |\Sigma_j|}} \exp\left[-\frac{1}{2}\mathbf{m}_j{}^T \Sigma_j{}^{-1} \mathbf{m}_j\right] \quad \text{Eq 7-24}$$

In the EM-framework, the E-step involves finding the joint probabilities of the complete data given the current estimate of the parameters, that is:

$$q(k,j) = q(\mathbf{x}_k, \mathbf{y}_j) = \alpha_j p_j(\mathbf{x}_k|\mathbf{0}_j) = \alpha_j p_j(x_k|\mathbf{m}_j, \Sigma_j) \quad \text{Eq 7-25}$$
$$= \frac{\alpha_j}{\sqrt{(2\pi)^D |\Sigma_j|}} \exp\left[-\frac{1}{2}\mathbf{m}_j{}^T \Sigma_j{}^{-1} \mathbf{m}_j\right]$$

In the mixture model problem, it is useful to also evaluate the a-posteriori probabilities of $p(j|k)$

$$p(j|k) = \frac{q(k,j)}{\sum_{i=1}^{K} q(i,j)} \quad \text{Eq 7-26}$$

Having found the set of conditional probabilities $p(j|k)$, the M-step of the EM algorithm is used to update the estimates of the parameters:

$$\mathbf{m}_j = \frac{\sum_{k=1}^{K} p(j|k)\mathbf{x}_k}{\sum_{k=1}^{K} p(j|k)} \quad \text{Eq 7-27}$$

$$\Sigma_j = \frac{\sum_{k=1}^{K} p(j|k)(\mathbf{x}_k - \mathbf{m}_j)(\mathbf{x}_k - \mathbf{m}_j)^T}{\sum_{k=1}^{K} p(j|k)} \quad \text{Eq 7-28}$$

The hidden data set $\{\alpha_j\}$ can also be found as:

$$\alpha_j = \frac{1}{K} \sum_{k=1}^{K} p(j|k) \quad \text{Eq 7-29}$$

Equations Eq 7-25 to Eq 7-29 are iterated until the parameters $\{(\mathbf{m}_j, \boldsymbol{\Sigma}_j)\}$ converges within a tolerance, or the number of steps is reached.

This section derives the steps for the EM algorithm for the mixture model problem. The next section shows the steps to solve the problem of motion segmentation given the motion vector field.

7.2.2 The Basic EM-based Motion Segmentation Algorithm

Recall in the EM algorithm in finding parameters of the Gaussian mixture model in the previous section. Given a set of K iid observations points $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_K\}$ which are assumed to be induced by a mixture of J Gaussian distributions $\{(\mathbf{m}_1, \boldsymbol{\Sigma}_1), (\mathbf{m}_2, \boldsymbol{\Sigma}_2), \cdots, (\mathbf{m}_J, \boldsymbol{\Sigma}_J)\}$ whose mixture probabilities, $\{\alpha_1, \alpha_2, \cdots, \alpha_J\}$ forms the hidden data set.

To use the EM algorithm for motion segmentation, the observed data is redefined as $\{(\mathbf{v}_k, \mathbf{p}_k): \ k = 1,2,..., K\}$where k is the lexicographical block index, $\mathbf{v}_k$ the motion vector of block k and $\mathbf{p}_k$ the position vector of the centre of block k.

It is assumed that the motion vector of each block k is due to one of the J possible independent motion. Each motion is represented by a set of motion model parameters, $\mathbf{a}_j$. Thus the parameter set is $\{\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_J\}$ which needs to be estimated by the EM algorithm. The hidden data set is the segment proportion of the blocks having the j^{th} motion parameters, $\{p(j)\}$. This is very similar to the mixture probability of the mixture model problem. In fact $p(j)$ can be viewed as the probability that a block is moving with the motion specified by the parameter set $\mathbf{a}_j$. In the remaining part of the section, the affine motion model is used to represent the motion of each component. Hence there are J set of parameters, $\mathbf{a}_j$ each generating a global motion field $\{\mathbf{v}_{jk} : k = 1 \ldots K\}$. By using the generated global motion field, the probability that the observed motion vector in block k, $\mathbf{v}_k$ can be modelled as a Gaussian distribution:

$$e_j(k) = \left\| \mathbf{v}_k - \mathbf{v}_{j,k} \right\| \qquad \text{Eq 7-30}$$

$$\sigma_j^2 = \frac{1}{K} \sum_{k=1}^{K} e_j^2(k)$$

$$p(k \mid j) = \frac{1}{\sqrt{2\pi\sigma_j^2}} \exp\left(-\frac{e_j(k)^2}{2\sigma_j^2} \right)$$

Comparison with the Gaussian mixture model reveals that Eq 7-30 is a simplified version of Gaussian distribution where the dimensionality is reduced to one.

The a-posteriori probabilities $p(j|k)$ can likewise be calculated as:

$$p(j,k) = p(j)p(k \mid j) \qquad \text{Eq 7-31}$$

$$p(k) = \sum_{j=1}^{J} p(j)p(k \mid j) = \sum_{j=1}^{J} p(j,k)$$

$$p(j \mid k) = \frac{p(j)p(k \mid j)}{\sum_{i=1}^{K} p(j)p(i \mid j)} = \frac{p(j,k)}{p(k)}$$

As the parameter set of the motion segmentation is significantly different from that of the mixture model problem, the M-step is rightfully dissimilar. It is based on the regression method as shown below:

$$\begin{bmatrix} W_{j,1} & 0 & \cdots & 0 \\ 0 & W_{j,2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & W_{j,K} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_K \end{bmatrix} = \begin{bmatrix} W_{j,1} & 0 & \cdots & 0 \\ 0 & W_{j,2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & W_{j,K} \end{bmatrix} \begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ \vdots & \vdots & \vdots \\ x_K & y_K & 1 \end{bmatrix} \begin{bmatrix} a_{j,0} \\ a_{j,1} \\ a_{j,4} \end{bmatrix}$$

$$\begin{bmatrix} W_{j,1} & 0 & \cdots & 0 \\ 0 & W_{j,2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & W_{j,K} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_K \end{bmatrix} = \begin{bmatrix} W_{j,1} & 0 & \cdots & 0 \\ 0 & W_{j,2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & W_{j,K} \end{bmatrix} \begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ \vdots & \vdots & \vdots \\ x_K & y_K & 1 \end{bmatrix} \begin{bmatrix} a_{j,2} \\ a_{j,3} \\ a_{j,5} \end{bmatrix} \qquad \text{Eq 7-32}$$

$$W_{j,k} = p(j \mid k) \;\; ; \;\; \begin{bmatrix} u_k \\ v_k \end{bmatrix} = \mathbf{v}_k \;\; ; \;\; \begin{bmatrix} x_k \\ y_k \end{bmatrix} = \mathbf{p}_k$$

$$\begin{bmatrix} a_{j,0} & a_{j,1} & a_{j,2} & a_{j,3} & a_{j,4} & a_{j,5} \end{bmatrix}^T = \mathbf{a}_j$$

Lastly, the segment proportions can be evaluated with:

$$p(j) = \frac{1}{K} \sum_{k=1}^{K} p(j \mid k) \qquad \text{Eq 7-33}$$

It can be seen that Equations Eq 7-30 to Eq 7-33 constitute an iterative formulation. Given an initial set of segments parameters $\{\mathbf{a}_j\}$ and segment proportions $\{p(j)\}$, the a-posteriori $\{p(j|k)\}$, joint $\{p(j,k)\}$probabilities can be obtained. The EM process corresponds to finding the maximum likelihood of the segment parameters given the motion vector field, with the segment proportions the hidden variable. Hence it is a probability mixture problem. Figure 7.2 shows the data dependency graph where each balloon is the data and the arrows represent how the data are related. The top part of Figure 7.2

represented by the upper balloon broken line is the E-step of the EM-based motion segmentation; the bottom part is the M-part.

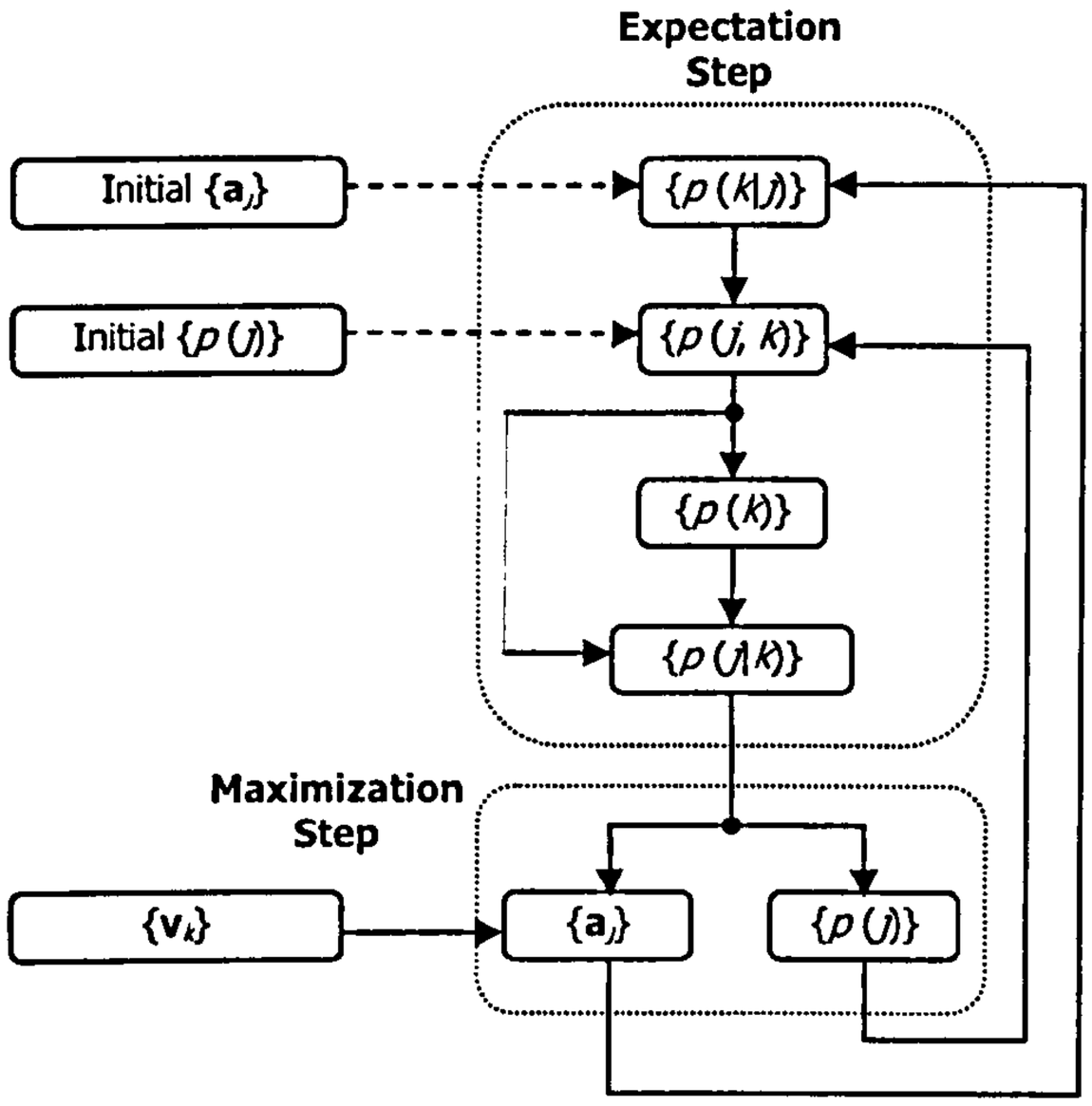


Figure 7.2 Block Description of the Basic EM-based motion segmentation process.

Each block n can then be assigned a segment membership $\{z_k\}$ as:

$$z_k = \arg\max_{j \in \{0,1,\ldots J-1\}} p(j \mid \mathbf{v}_k) \qquad \text{Eq 7-34}$$

The basic principle behind the EM-based motion segmentation is presented in this section. The following section provides a set of improvements are proposed to improve the basic method.

7.2.3 Details and Improvements in EM-Based Motion Segmentation

7.2.3.1 Similar Region Merging

Depending on the initial parameters and starting regions, there are instances where regions with similar parameters are segmented into more than one segments. Hence there is a need to identify and combine such regions. Two sets of motion parameters are considered similar if each parameter pair produces a displacement less than the resolution of the motion vector component at the corners of the picture. Considering affine models of two regions j_1 and j_2, $\{a_{0,j_1}, a_{1,j_1}, a_{2,j_1}, a_{3,j_1}, a_{4,j_1}, a_{5,j_1}\}$ and $\{a_{0,j_2}, a_{1,j_2}, a_{2,j_2}, a_{3,j_2}, a_{4,j_2}, a_{5,j_2}\}$, the regions are deemed to be similar and are to be merged if:

$$\sum_{p=0}^{5} s_p \left| a_{p,j_1} - a_{p,j_2} \right| < r \qquad \text{Eq 7-35}$$

where s_p is the scaling factors and r the similarity threshold. The translational scaling factors (s_4 and s_5) are set at 16 pixels; the other factors (s_0, s_1, s_2, s_3) are dependent on the picture size. More specifically, these factors are themselves scaled by half the value of the picture width or picture height, which ever is larger. For instance, for the value used in QCIF sequences is $88 \times 16 = 1048$ whereas that used in CIF sequences is $176 \times 16 = 2096$. If the sum of the scaled value is less then one, we make the decision to merge the two segments together, as there is no difference between the fields generated by the two parameter sets, the two regions are merged.

The merging process is carried out after the motion parameter estimation step. When segments j_s and j_t are considered similar and to be merged, one segment is chosen arbitrarily to be merged into the other. For discussion purposes we shall assume j_t is merged with j_s. During the merging process, the following parameters are changed according to

$$\begin{cases} p(j_t) \rightarrow p(j_t) + p(j_s) \\ p(j_s) \rightarrow 0.0 \end{cases} \qquad \text{Eq 7-36}$$

$$\forall k = 1 \cdots K: \quad \begin{cases} p(j_t \mid k) \rightarrow p(j_t \mid k) + p(j_s \mid k) \\ p(j_s \mid k) \rightarrow 0.0 \end{cases}$$

This algorithm is found to be crucial in the EM-based motion segmentation as it removes redundant segments from the iterative process, thus speeding up the convergence rate. By using this similar region merging (SRM), the initial number of segments can be made arbitrarily big as they will be merged eventually. Of course the larger the initial number, the more iterative steps are required before the desired convergence tolerance is met. In the simulations below, the initial segmentation is a 4x4 partition.

7.2.3.2 Iteration Stopping Criteria

As for any iterative algorithm, there should be some termination criterion. This is usually done to determine when the solution converges to a desired accuracy, and safe-guarded by a maximum number of iterations so that the process will always terminate. Some implementations also detect signs of divergence which brings the solution progressively away from the actual solution.

In the proposed implementation, convergence is determined by a similar measure as that used in the similar region merging (SRM) algorithm described in Eq 7-35. Instead of comparing parameters between two segments, the parameters of every segment are compared with the corresponding

parameters of the previous step. If the sums are less than unity for all the non-empty segments, convergence is reached and the process terminates.

In the basic EM methods, the algorithm takes between 140 to 200 iterations before convergence, depending on the initial segmentation and parameters. As a safeguard, an ultimate limit to the number of iteration of the EM algorithm can undergo is imposed. If the number of iterations reaches 256, the process is terminated and the last set of model parameters and mixture probabilities are taken as the final estimate.

7.2.3.3 Observation Data Adaptation via Candidate Points

The result of motion estimation is a motion vector field, which forms the observation data set for the EM-process. The field itself may contain inaccuracies due to the generalized aperture problem. This thesis makes an improvement to the EM process by using separate fields for each segment. This is based on the set of candidate points discussed in Chapter 4 and is used in a similar way as in Chapter 6.

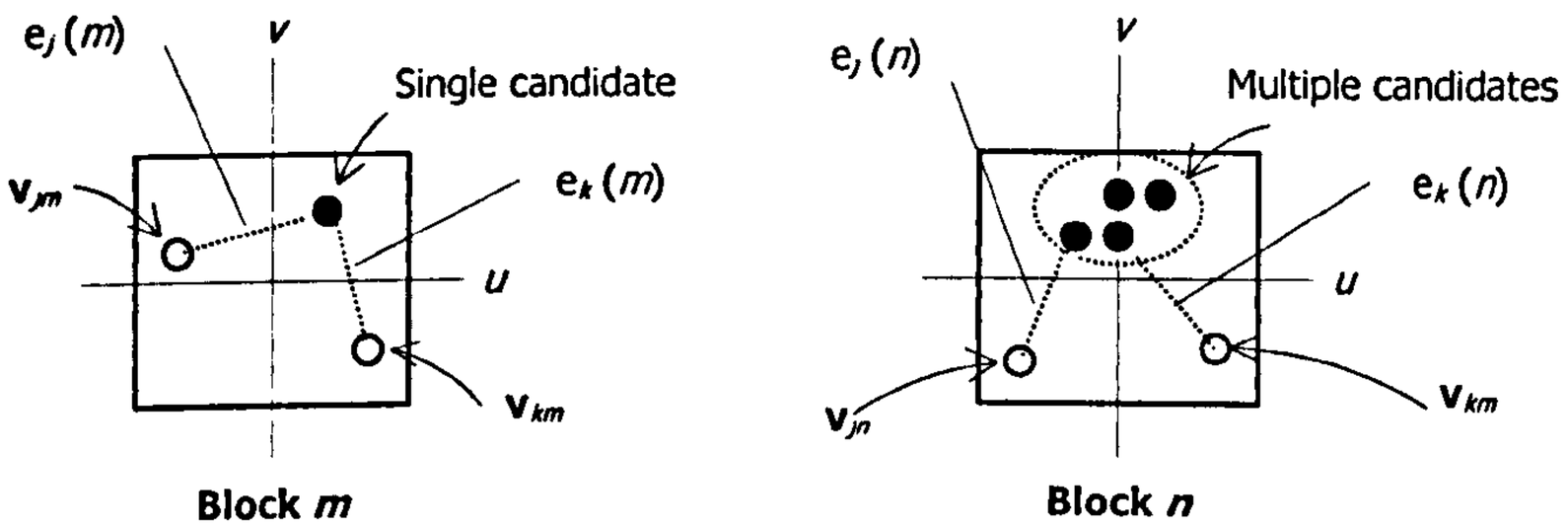


Figure 7.3 Diagram demonstrating adaptive changing of observation point with segment parameters.

Figure 7.3 shows two blocks and the dark dot in block m shows a single candidate point. This candidate point is the motion vector observed in block m. This block has a good motion vector as it has only one single candidate motion vector. The white dots are motion vectors of this block due to the motion parameters of two segments, j and k. On the other hand, block n has a few candidate vectors whose SADs are all low enough to be qualified as motion vectors. By adaptively choosing the closest candidate points to the segments' motion vector as observation points, the EM motion segmentation algorithm can approach towards the true parameters even in the presence of the general aperture problem. The adaptation scheme is called candidate vector adaptation (CVA) and it will be shown to increase convergence rates as well as improve segmentation accuracies.

In addition to the improvements proposed in the last section to the basic EM-based motion segmentation, pre-processing and post-processing algorithms are proposed to further improve the convergence and compression capability of the segmentation algorithm.

To distinguish between the algorithms with and without the pre- and post-processing, the former is termed the Adaptive EM-based segmentation (AEMS, owing to CVA described in the previous paragraph). The latter is termed the AEMS with Pre- and Post-processing (PPAEMS).

7.2.4 Segment Initialization via Hough Transform

In AEMS, the initial segmentation is a 4×4 partition of the picture (to be shown in the following section on simulation results). The mixture probabilities $\{p(j)\}$ are then initialized to 1/16 and an every partition is global motion estimated to obtain the initial model parameter estimates $\{\mathbf{a}_j\}$. This is one of the two common practice used in iterative motion segmentation algorithms, the other being producing an initial partition via texture-, or colour- or simple greyscale-based segmentation.

This thesis proposes an extension of PHGME to provide an initial estimates of $\{p(j)\}$ and $\{\mathbf{a}_j\}$. The process, termed successive progressive Hough Transform-based motion segmentation (SPHMS), involves iteratively performing a modified version of PHGME on the motion vector field obtained by QBMA. After each step of the iteration, the blocks whose local motions match that of the global motion within the PHGME resolution threshold is eliminated from the subsequent iteration. By doing so, more dominant segments are isolated before the less dominant ones. As each stage eliminate the current segments from the next estimation, processing time decreases at every stage. Furthermore, the number of segments is automatically identified as the observation data get exhausted.

In contrast to the PHGME, estimation accuracy is not of paramount importance in SPHMS, as it is used to obtain an initial segment for subsequent segmentation algorithm; the processing time is a more crucial factor. Furthermore, subsequent dominant motion is more likely to be affine than translational+zoom. Taking these factors into consideration, SPHMS uses only 3-parameter translation-zoom model, and uses a small accumulator size of $9\times9\times9$. With an iterative count of three and initial parameter extent of $\{\pm\frac{1}{8},\pm16,\pm16\}$, the final resolution of the SPHMS parameters are $\{\pm\frac{1}{512},\pm\frac{1}{4},\pm\frac{1}{4}\}$. By applying SPHMS to obtain the initial parameters prior to AEMS, the chances of arriving at the global optimal result are greatly enhanced. With SPHMS, the number of iterations for AEMS to converge is also greatly reduced to 30-60.

7.2.5 Insignificant Segment Elimination and Outlier Detection

After AEMS with SPHMS pre-processing, the segmentation result typically contains around twelve to fourteen segments with a small set of one to four dominant segments. The remaining segments are insignificant and can be eliminated. The process of insignificant segment elimination (ISE) is as follows. Assuming there are J segments and the final mixture probabilities are $\{p(j): j=0,\ldots,J-1\}$.

The segments whose mixture probability is below $1/J$ are deemed insignificant and eliminated from subsequent processing. The remaining segments forms significant mixture set, S_J:

$$S_J = \left\{ j \in [0,\ldots,J-1] : p(j) > \frac{1}{J} \right\} \qquad \text{Eq 7-37}$$

A subsequence of ISE is possibility of identifying the small amount of outliers which do not belong to any member of S_J. The outlier set O_K contains members of the observation sets (blocks 0, …, K–1) where blocks' maximum memberships do not belong to the significant mixture set

$$O_K = \left\{ k \in [0,\ldots,K-1] : \tilde{z}_k \notin S_J \right\} \qquad \text{Eq 7-38}$$
$$\tilde{z}_k = \arg\max_{j=0,\ldots,J-1} p(j|k)$$

The concept behind ISE, S_J and O_K are illustrated in

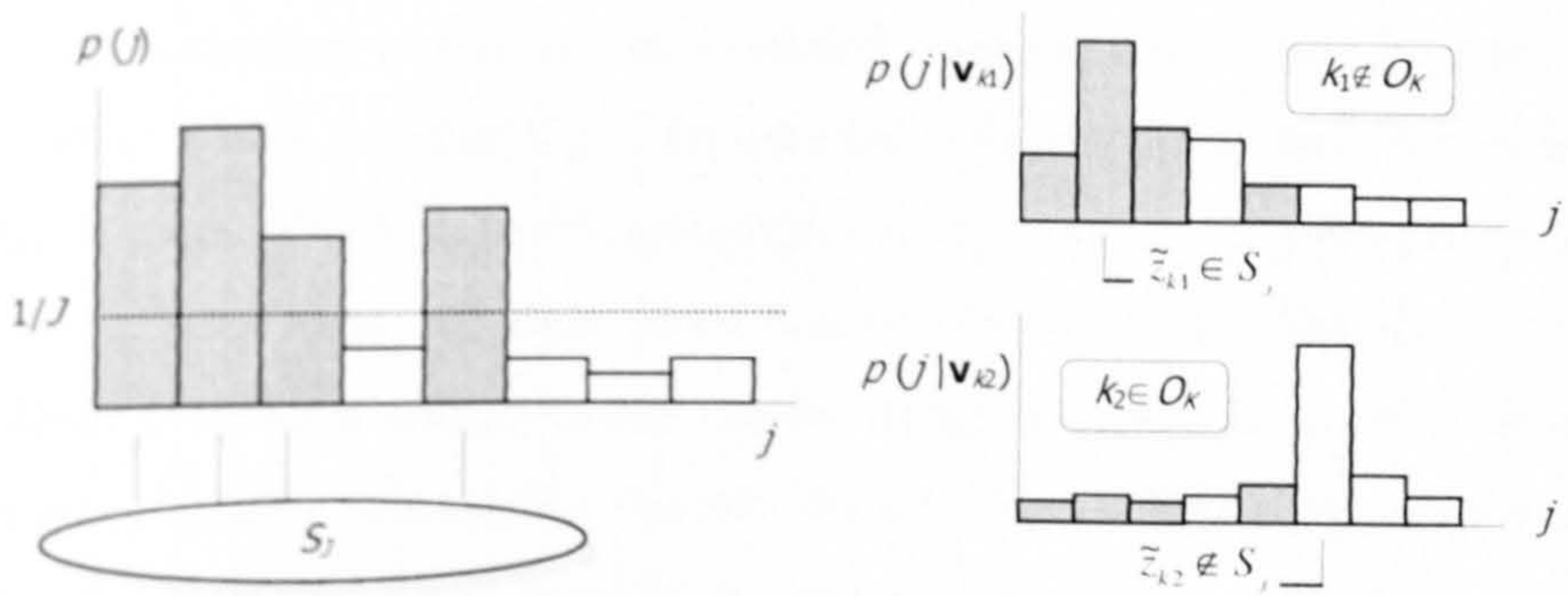


Figure 7.4 Diagram demonstrating the concepts of insignificant segments elimination, significant set and outlier set.

7.2.6 Queue-based Segmentation Simplification

Although AEMS produces a low entropy motion field, the segmentation map is rather fragmented in nature, as shown in Figure 7.5. Regardless of which coding schemes used, such a fragmented segmentation map requires excessive bits to code. In this section an algorithm similar to that of QBMA is proposed to simplify this map. In this proposed method called queue-based segmentation simplification (QSS), a priority queue is first set up according to the 'reliability' of the block's segment assignment. 'Reliable' blocks are processed first, using the maximum membership principle represented in Eq 7-34. When a subsequent block is processed and a neighbouring block is already assigned a segment, the current membership probabilities of the current block are modified to weigh

towards its processed neighbour. The segment assignment is then carried out using the modified membership probabilities.

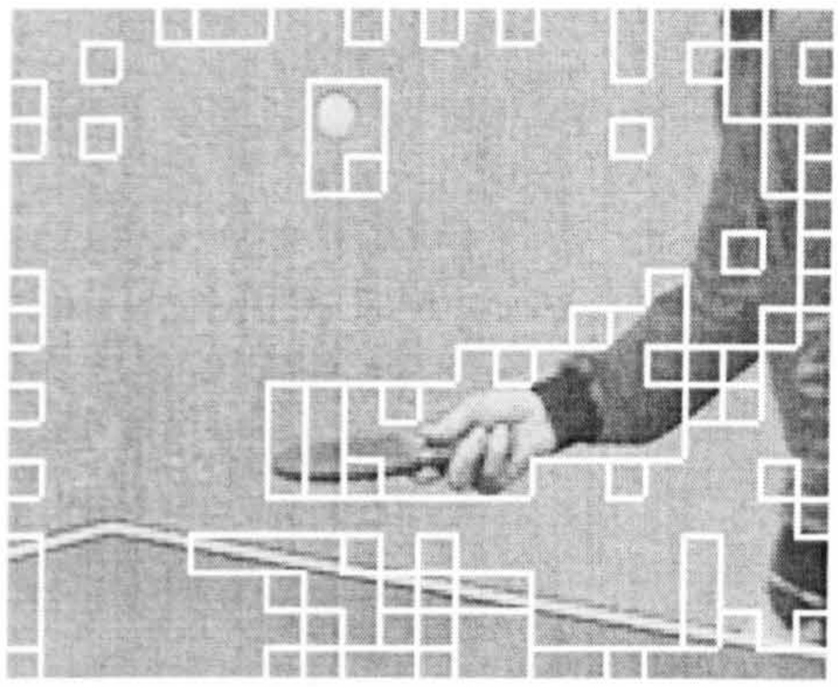


Figure 7.5 Segmentation map of a frame resulting from AEMS.

Prior to QSS, blocks belonging to the outlier set O_K are excluded from the process, and the membership probabilities of the remaining blocks are recalculated with the significant set (S_J). It is noted that entropy of each block's new membership is an excellent reliability measure. Referring to Figure 7.6, the left block has a more reliable segment assignment as it contains a significant peak in the mixture probabilities. On the other hand, the right block has no obvious peak. The entropy of the former is lower than the latter's. Hence a priority queue can be set up in ascending order of the block's mixture entropy. In that way, 'reliable blocks' like the left block in Figure 7.6 can be processed first and non-reliable ones like the right block can modify its mixture probabilities according to its more reliable neighbours.

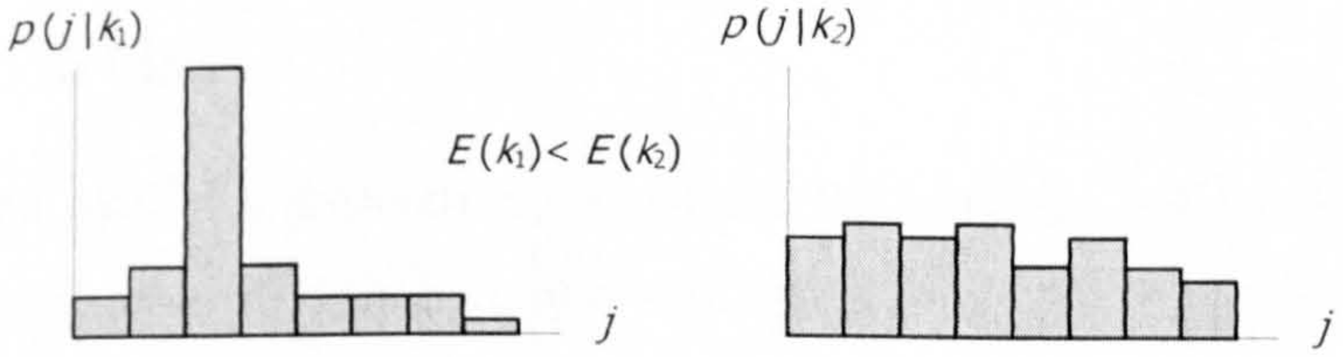


Figure 7.6 Diagram showing the mixtures entropies of two blocks. The left block is more reliable than the right block.

Mixture modification involves averaging the current modified mixture $\{p(j|k_1)\}_j$ with another mixture $\{p(j|k_2)\}_j$:

$$p(j|k_1,k_2)=(1-\frac{r_{k_1k_2}}{2})p(j|k_1)+\frac{r_{k_1k_2}}{2}p(j|k_2) \qquad \text{Eq 7-39}$$

The relative weight $r_{k_1k_2}$ denotes how much influence the neighbouring block k_2 has on the current block k_1. An ideal measure of $r_{k_1k_2}$ is the Bhattacharya distance:

$$r_{k_1k_2}=\sum_{j\in S_j}\sqrt{p(j|k_1)p(j|k_2)} \qquad \text{Eq 7-40}$$

The Bhattacharya similarity measure is used to quantify the similarity between two distributions and varies between 0 and 1. The membership mixture of the neighbouring block is 'mixed' with the current block's mixture; the more two the distributions resemble each other the higher the value of $r_{k_1k_2}$. By comparing the revised membership of the current block with all its processed neighbours, maximum component is identified as the current segment:

$$z_k=\arg\max_{j\in S_j,k\in N_k} p(j|k,k') \qquad \text{Eq 7-41}$$

In Eq 7-41, N_k stands for the set of blocks which are neighbours of block k and is above k in the priority queue. By means of the priority queue and the membership revision, QSS produces are less fragmented segmentation map.

7.3 Simulation Results

7.3.1 Synthetic fields

To test the EM-based motion segmentation's accuracy, the synthetic motion vector field used in the previous chapter is used for ground truth comparison: a QCIF image with motion vectors based on 4x4 blocks (Figure 6.12 and Table 6.1).

With the proposed EM-based motion segmentation, using a starting segmentation of 16 square blocks, it takes six iterations to converge within accuracies of $\{\pm\frac{1}{88\times16},\pm\frac{1}{88\times16},\pm\frac{1}{88\times16},\pm\frac{1}{88\times16},\pm\frac{1}{16},\pm\frac{1}{16}\}$.

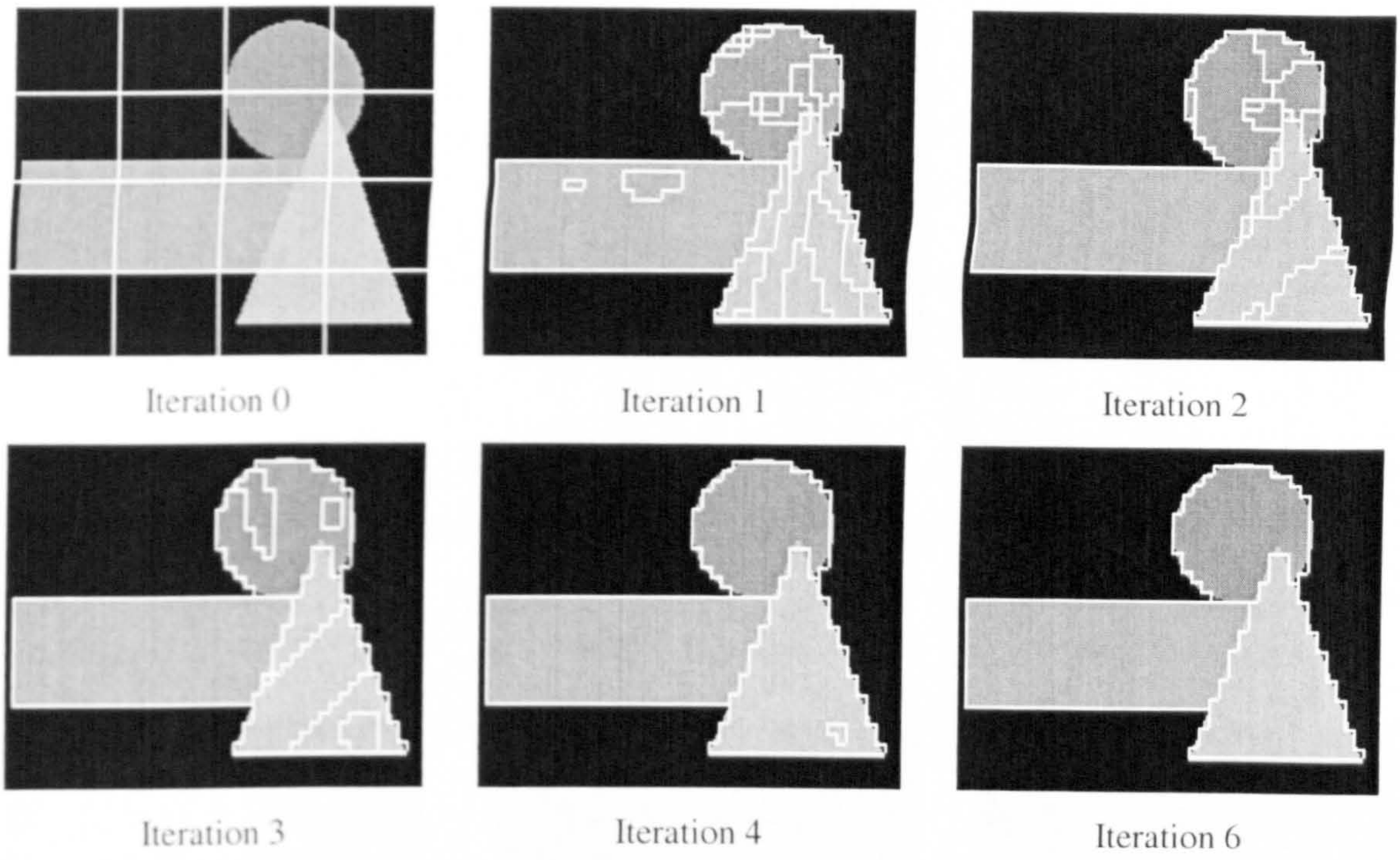
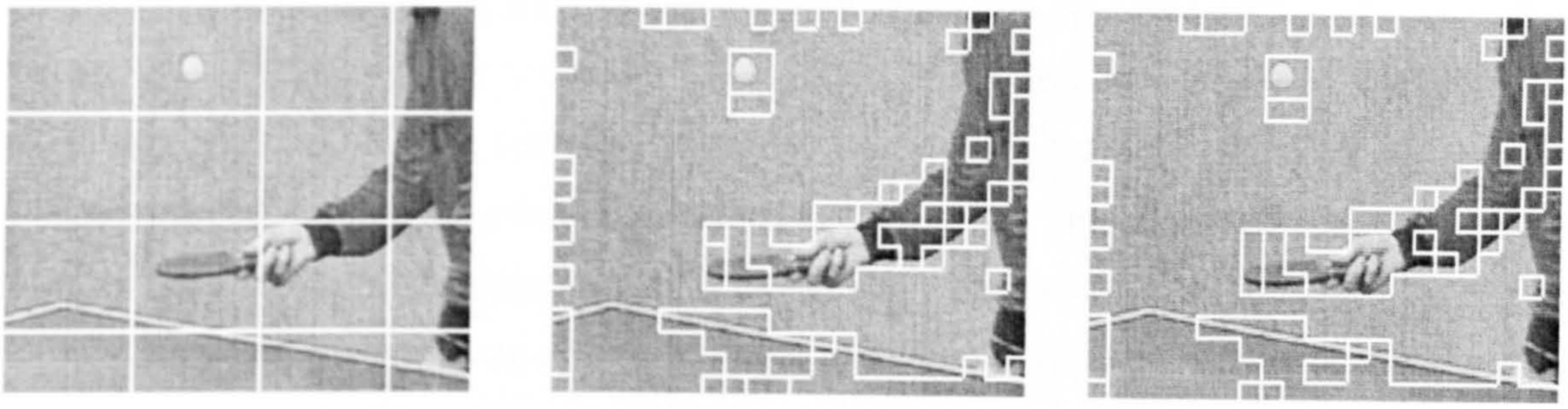


Figure 7.7. Segmentation maps of EM-based motion segmentation It takes 6 iterations to extract the 4 segments correctly.

As evident from Figure 7.7, all four segments can be accurately identified. It should be noted that with the test sequences, convergence is not necessary as rapid. The follow simulation results shows the convergence rate of the simple EM-segmentation on natural sequences and how candidate vectors adaptation can improve both the convergence rate and the segmentation accuracy.

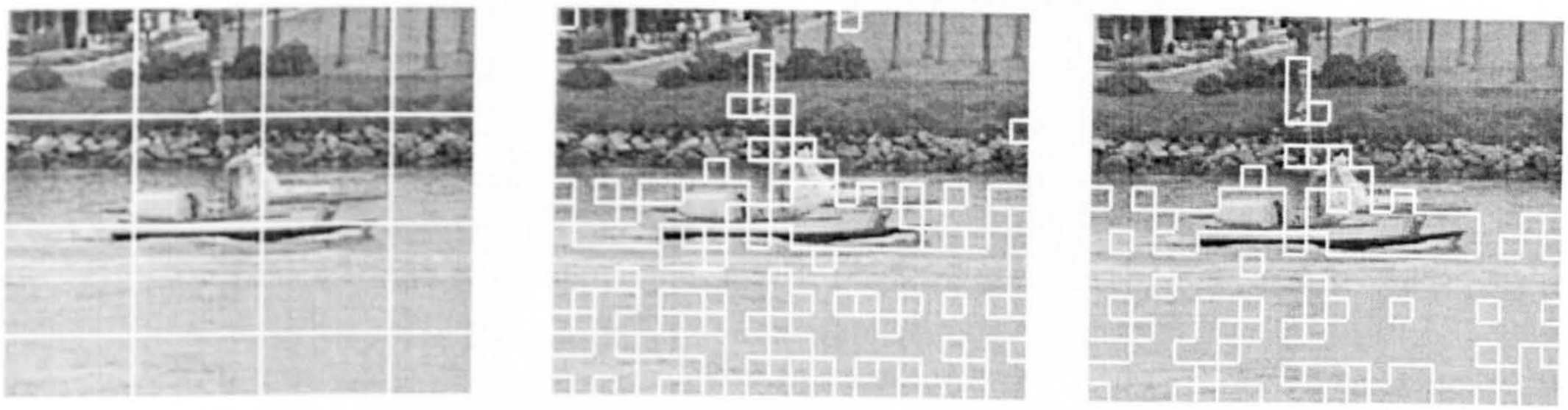
7.3.2 EM Convergence of Test Sequences with PPAEMS

The convergence of the basic EM-based segmentation for test sequence is not as rapid as that of the synthetic sequence. A typical frame takes around 90 to 200 iterations to converge to a within accuracies of 1/16 pixels, as shown in Figure 7.8, Figure 7.9 and Figure 7.10. In addition to the slow convergence, the final segmentation is rather fragmented and may be improved in various ways. The remaining of this section investigates and discusses the steps used to improve the EM-based motion segmentation algorithm as stated in the preceding sections.



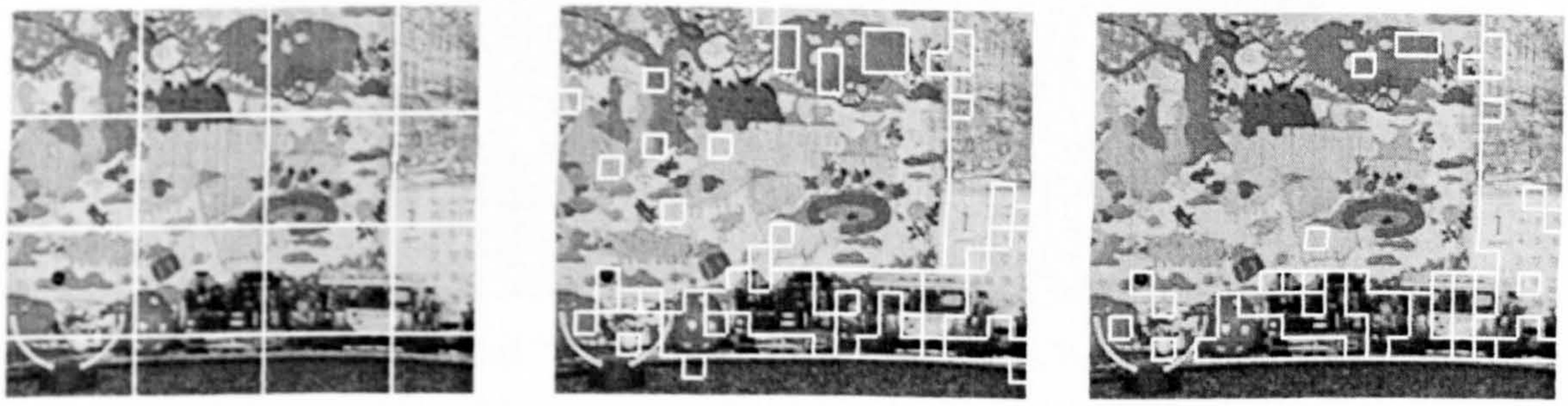
Iteration 0 Iteration 70 Iteration 140

Figure 7.8. Segmentation maps of EM-based motion segmentation on frame 33 of TABLE.QCIF (reference frame 30). It takes 140 iterations to stabilize to the 5 segments



Iteration 0 Iteration 75 Iteration 125

Figure 7.9. Segmentation maps of EM-based motion segmentation on frame 177 of COAST.QCIF (reference frame 174). It takes 125 iterations to stabilize to the 4 segments

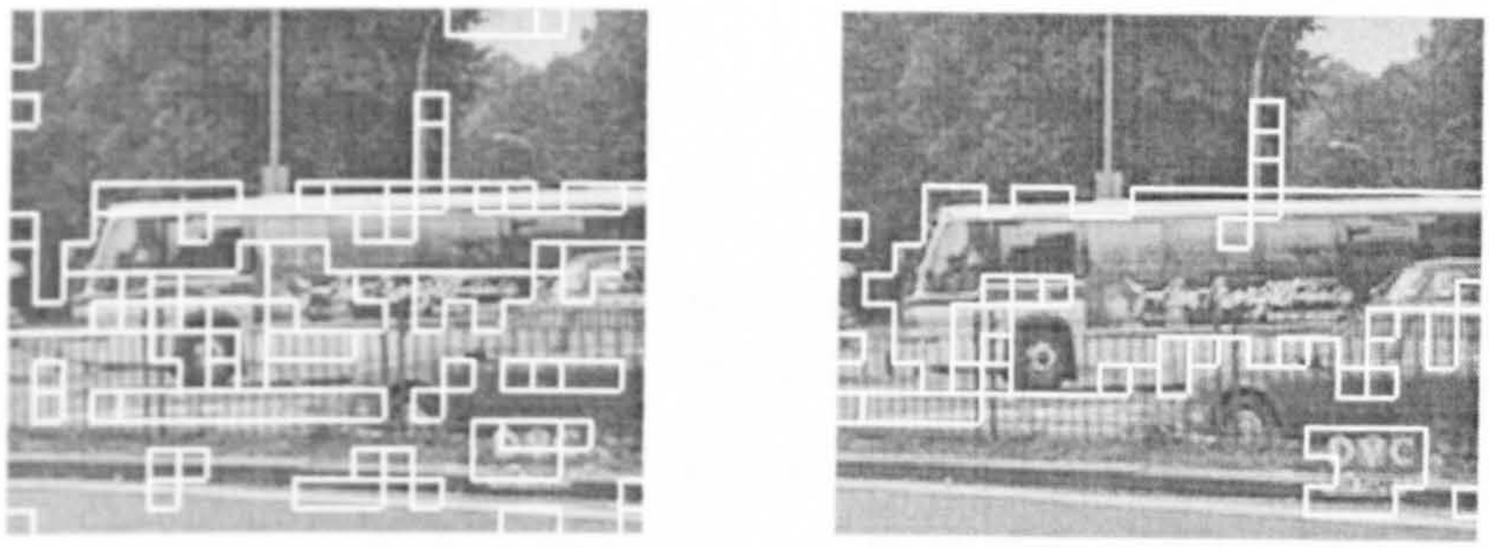


Iteration 0 Iteration 105 Iteration 231

Figure 7.10. Segmentation maps of EM-based motion segmentation on frame 234 of MOBILE.QCIF (reference frame 231). It takes 126 iterations to stabilize to the 6 segments

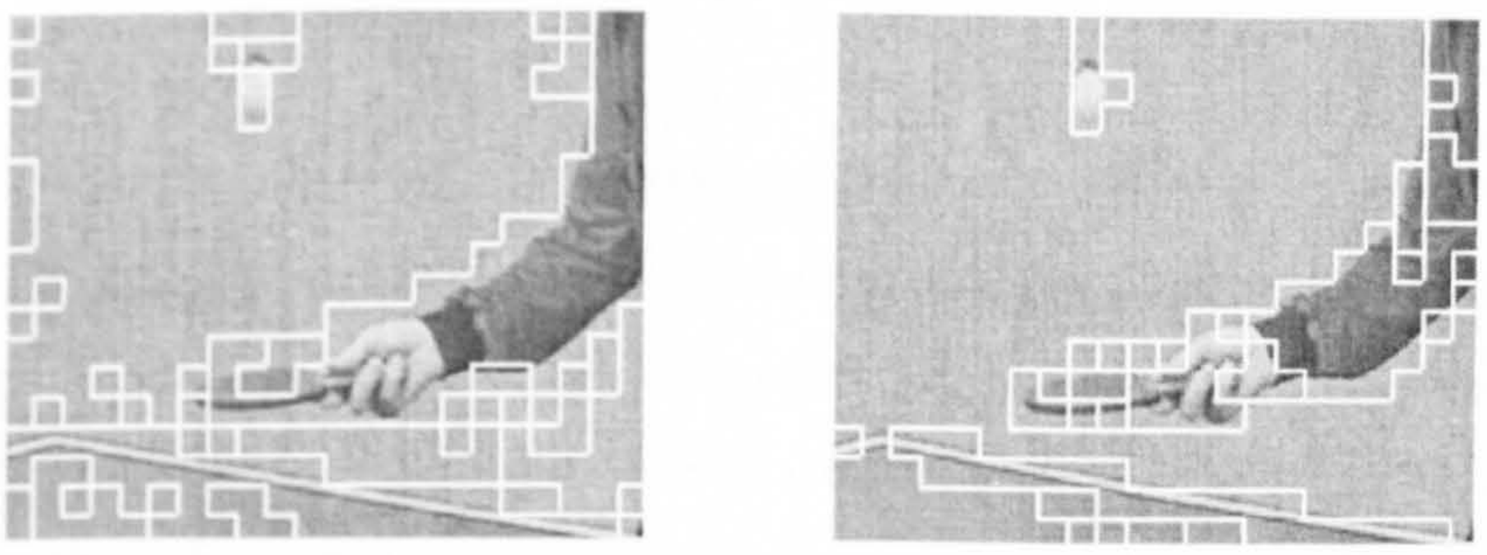
Firstly the use of candidate vector adaptation (CVA) is investigated. By allowing the EM algorithm to select the motion vectors amongst the candidate set for each block and for every segment, the block can

choose a motion vector field which best matches the field produced by the segment's parameters. The effect of CVA is illustrated in Figure 7.11 and Figure 7.12. As can be seen by comparing the left and right panes of both figures, the CVA-based EM produces a segmentation map more consistent with moving object boundary. The big bus and the static letterings at the bottom right corner in Figure 7.11 are less fragmented in the right panel (CVA-based EM MotSeg). The background in Figure 7.12 is more correctly segmented in the right pane; this appropriately illustrates the ability of CVA to segment a relatively textureless more effectively. Nevertheless, the segmentation at moving boundaries is still very fragmented, and the CVA does not solve the problem of convergence rate. In most cases, the number of iterations increases marginally with CVA. The use of Hough transform for segment initialization aims to solve these two problem.



(a) Standard EM-MotSeg (b) EM-MotSeg with CVA

Figure 7.11. Segmentation maps of EM-based motion segmentation on frame 246 of BUS.QCIF. (a) Standard EM; (b) EM with CVA.



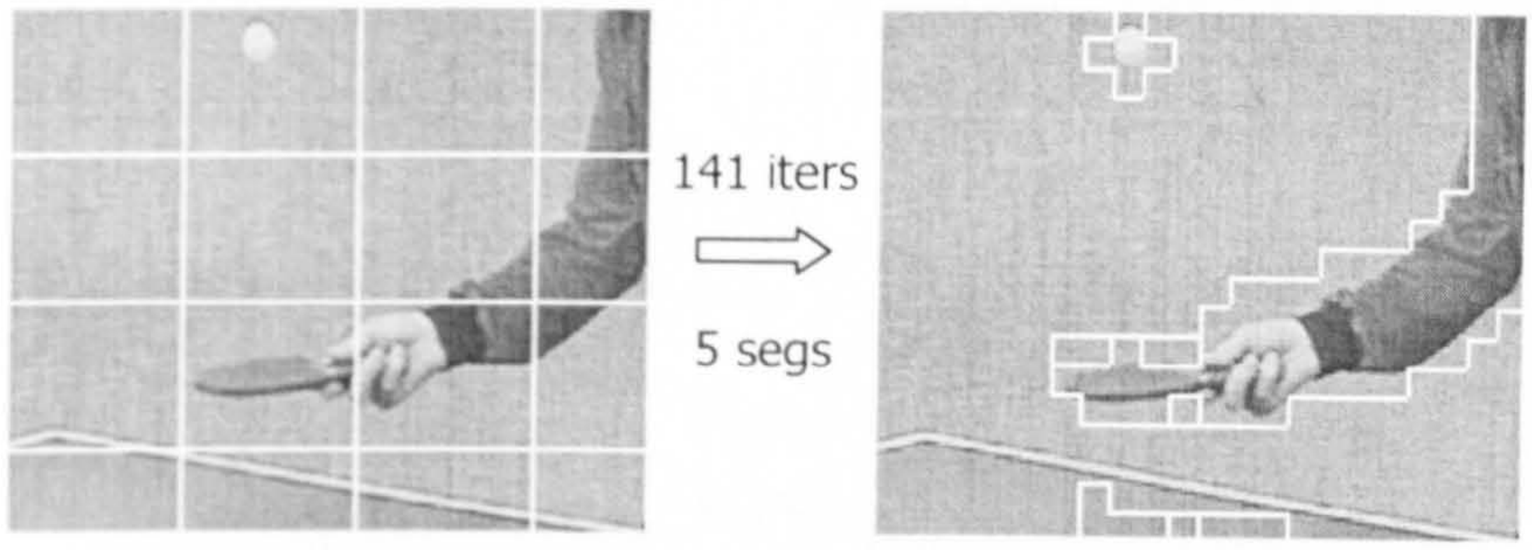
(a) Standard EM-MotSeg (b) EM-MotSeg with CVA

Figure 7.12 Segmentation maps of EM-based motion segmentation on frame 21 of TABLE.QCIF. (a) Standard EM; (b) EM with CVA.

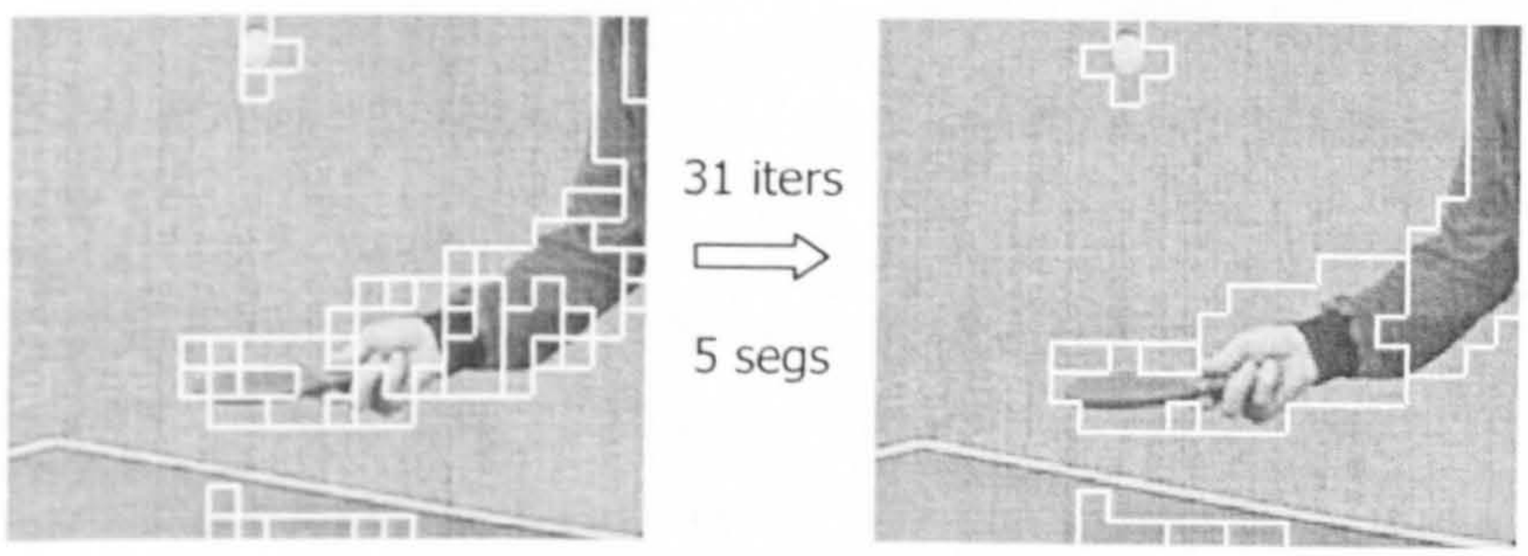
The standard EM uses an initial segmentation of 4x4 square regions, which is hardly the natural segmentation. By using Hough Transform-based (SPHMS) with the candidate vectors adaptation (CVA) algorithm, convergence rate has increased and the average of 30-60 iteration steps is achieved.

The segmentation results are also more consistent with the moving foregrounds and background, leading to a lower bit-rate requirement on the segmentation map.

As can be seen in Figure 7.13, the segmentation results of the 141st iteration of the simple EM-segmentation is not as good as the 31st iteration of EM-segmentation with candidate vectors adaptation.



(a)



(b)

Figure 7.13. Segmentation maps of EM-based motion segmentation on frame 33 of TABLE.QCIF. (a) With simple AEMS; (b) AEMS-segmentation with SPHMS. The left panels are the initial segmentations and the right are the final segmentations.

Another illustration of the superiority of *SPHMS-AEMS* over *AEMS* is shown in Figure 7.14. The number of iterations to achieve similar segmentation reduces from 203 in *AEMS* to 44 in *SPHMS-AEMS*. In addition, SPHMS is also able to reduce the number of segments – this is due to the fact that Hough transform detects the global maximum. Unlike the SPHMS, the standard EM MotSeg starts with an initial segmentation which may produce two segments within separate regions of the field, which attributed by 2 local minimums. The successive peak detection in SPHMS will be able to identify this two similar regions and product a single region. The SPHMS pre-processing improves the convergence rate of AEMS and increases the chances of reaching a globally optimal segmentation. However, the SPHMS-AEMS algorithm did not take spatial smoothness into consideration. The remainder of this section investigates the use of Queue-based Segmentation Simplification (QSS) as a post-processing algorithm SPHMS-AEMS to simplify the segmentation. Incorporation of SPHMS-based pre-processing and QSS-based post-processing to adaptive EM motion segmentation (AEMS) is termed Pre- and Post-processing AEMS (PPAEMS).

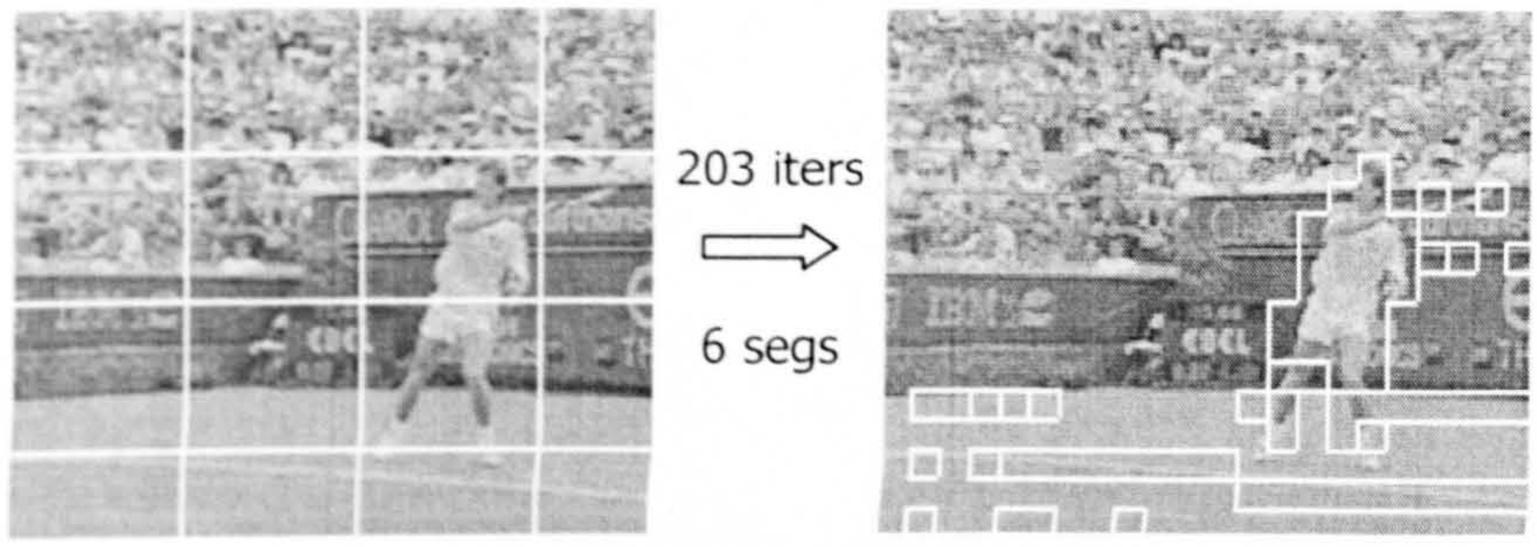


Figure 7.14. Segmentation maps of EM-based motion segmentation on frame 118 of STEFAN.QCIF. (a) With simple AEMS; (b) AEMS-segmentation with SPHMS. The left panels are the initial segmentations and the right are the final segmentations.

The QSS algorithm of PPAEMS simplifies the segmentation without incurring excessive increase in motion and texture entropies.

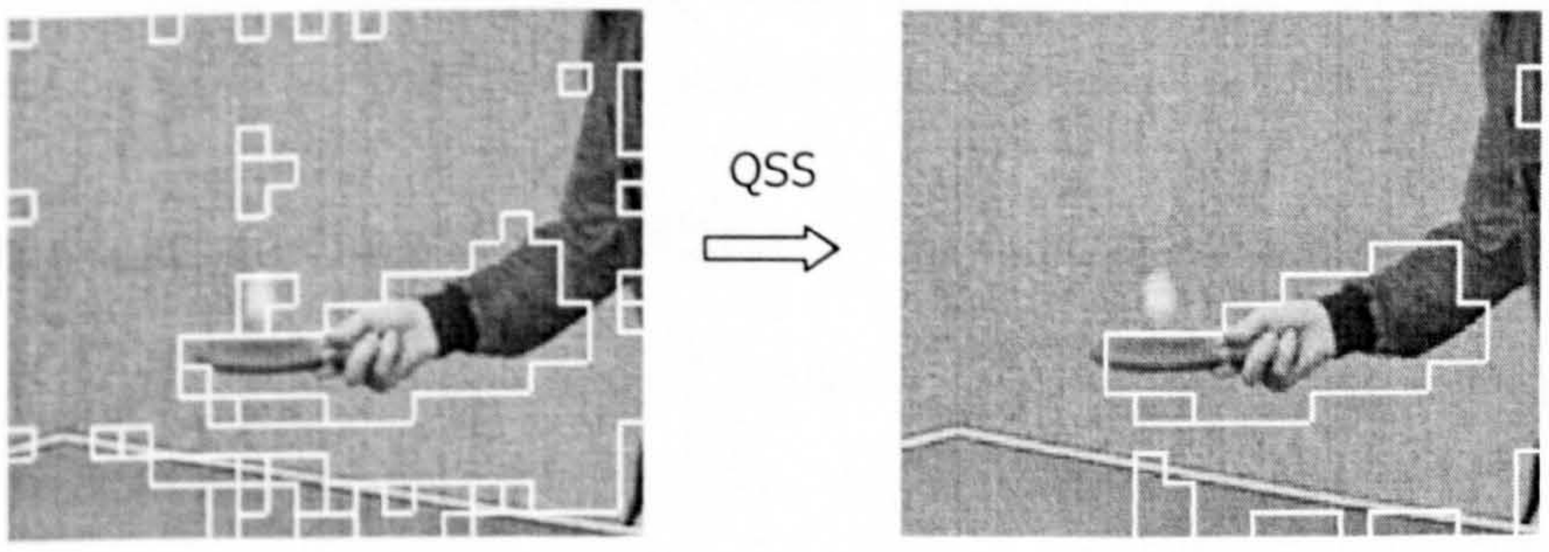


Figure 7.15. Result of QSS on frame 26 of TABLE.QCIF. Left panel is the segmentation result of SPHMS-AES and right panel is the result of applying QSS on the former.

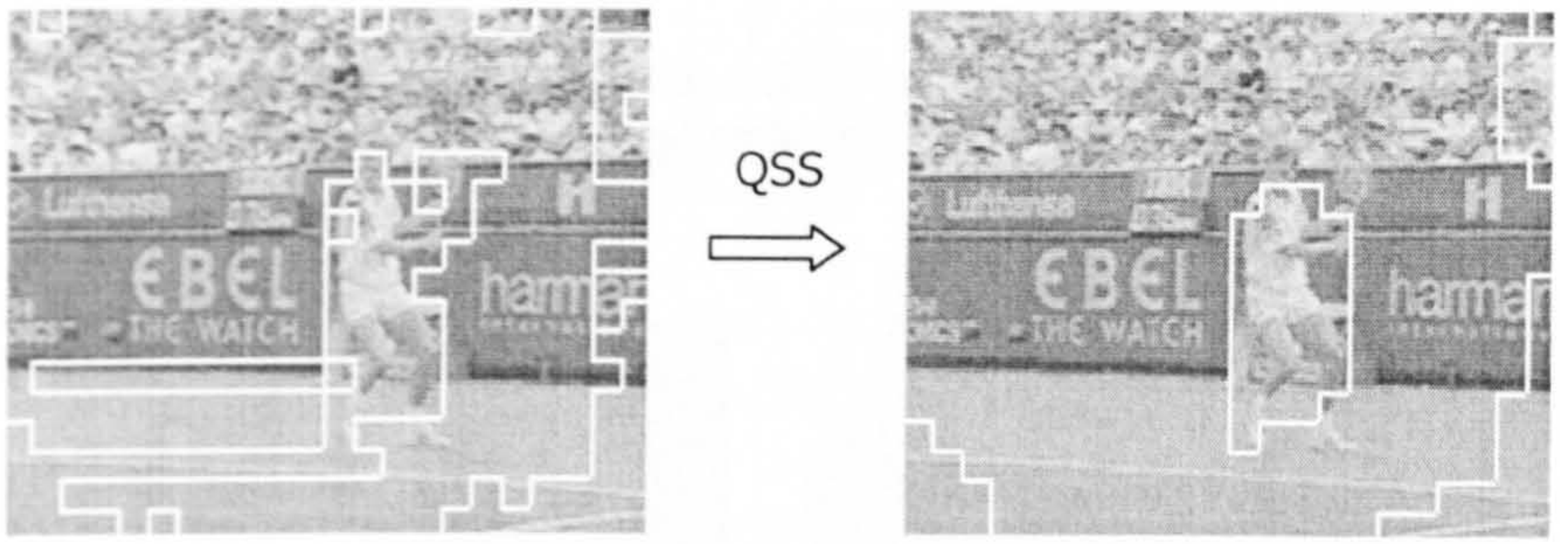


Figure 7.16. Result of QSS on frame 6 of STEFAN.QCIF. Left panel is the segmentation result of SPHMS-AES and right panel is the result of applying QSS on the former.

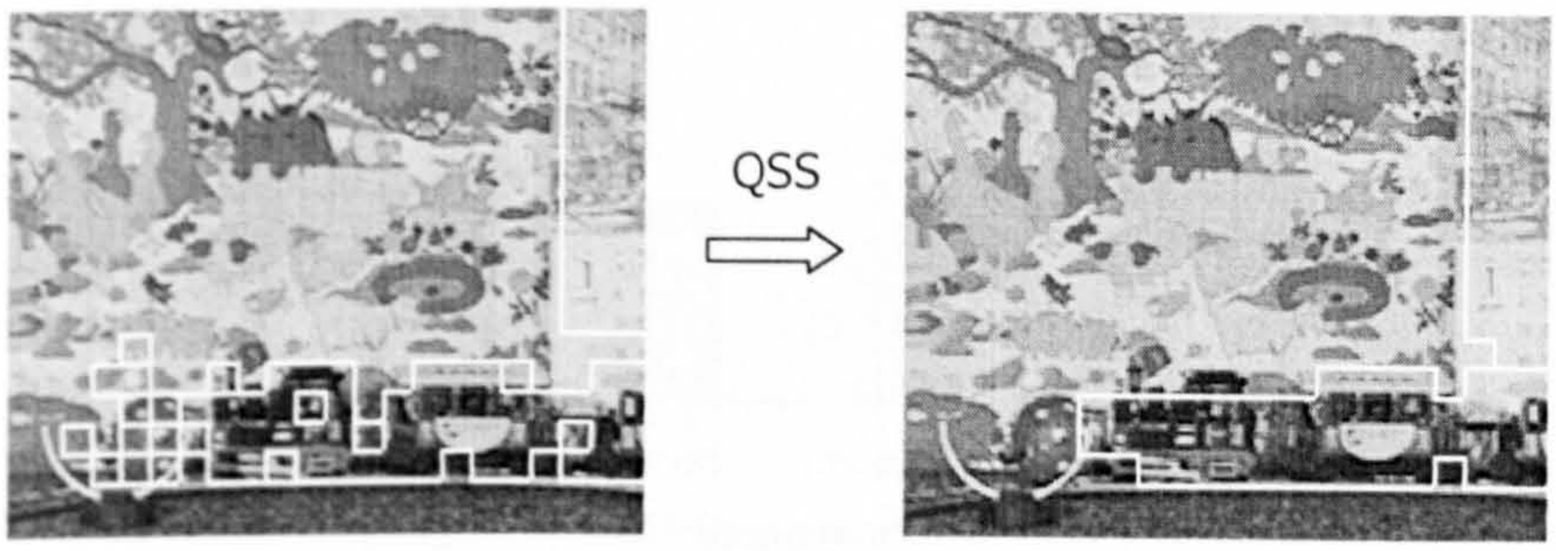


Figure 7.17. Result of QSS on frame 270 of MOBILE.QCIF. Left panel is the segmentation result of SPHMS-AES and right panel is the result of applying QSS on the former.

In the next two sections, simulation results show that motion vector fields can be more compactly represented using PPAEMS. The total entropy of a picture can also be reduced by using multiple warped versions of the reference frames using parameter sets evaluated with the PPAEMS algorithm.

7.3.3 Motion Compactness Capabilities of EM-Segmentation

The complete PPAEMS algorithm was applied to the six CIF and QCIF test sequences to test how much entropy can be removed from their respective motion vector fields. The results are depicted in Figure 7.18, whose legend represents the following algorithms:

1. dgmv: residual vector fields from SIRGME and residual motion vectors;
2. dsgmv: the motion vector entropy by PPAEMS.

Other than TABLE.CIF, all the sequence benefits from the use of PPAEMS over SIRGME; all sequences produce much lower motion entropies with PPAEMS than with QBMA. This is mainly attributed to the ability of PPAEMS to produce a more accurate estimate of the parameters for each segment. TABLE.CIF, consisting mainly of a single zoom factor, can be effectively removed by global motion estimation.

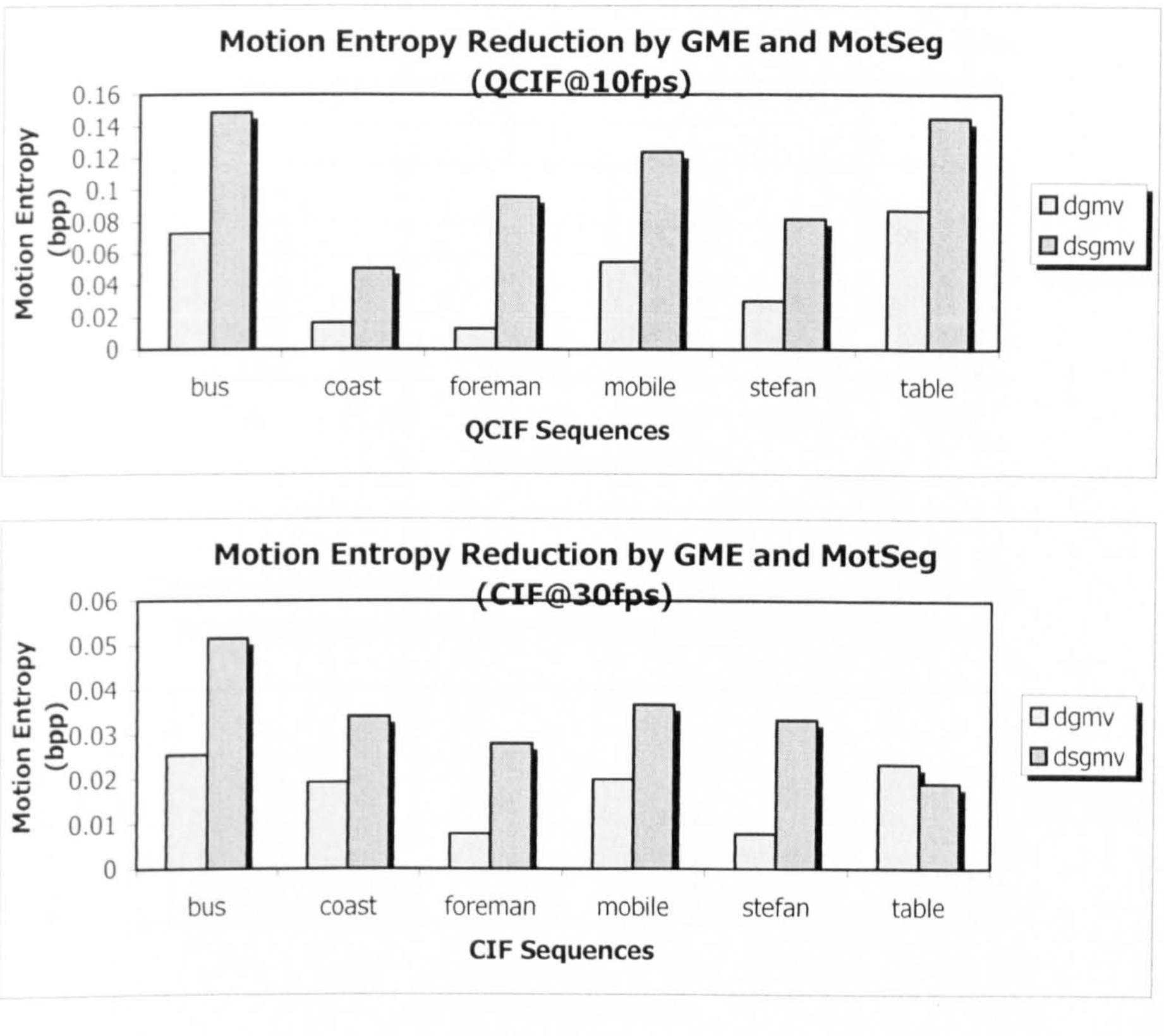


Figure 7.18. Charts showing the reduction of motion entropies by GME and MotSeg.

7.3.4 Using Motion Segmentation for Reference Picture Warping

Similar to the GME, warping reference frames provides a closer match to the input frame. By providing multiple warped versions according to the segmentation results and selecting the version best matches the local regions can even provide better results. Simulations on test sequences of the PPAEMS-QBMA pair are shown alongside the plain QBMA and SIRGME-QBMA algorithms in Figure 7.19, where:

1. wgme: represents the combined entropy reduction of QBMA with warped reference frame using parameters obtaiend from PHGME;
2. wsgme: represents the combined entropy reduction of QBMA with more than one warped versions of the reference frame using parameters obtaiend from PPAEMS.

The performance of PPAEMS is only marginally better then PHGME in terms of providing a better matched reference frame for the subsequent QBMA. The main reason is the excessive bits required to code the segmentation map, which not required in the case of PHGME.

The performance of PPAEMS will be more marked when a suitable compression scheme is found for the segmentation map.

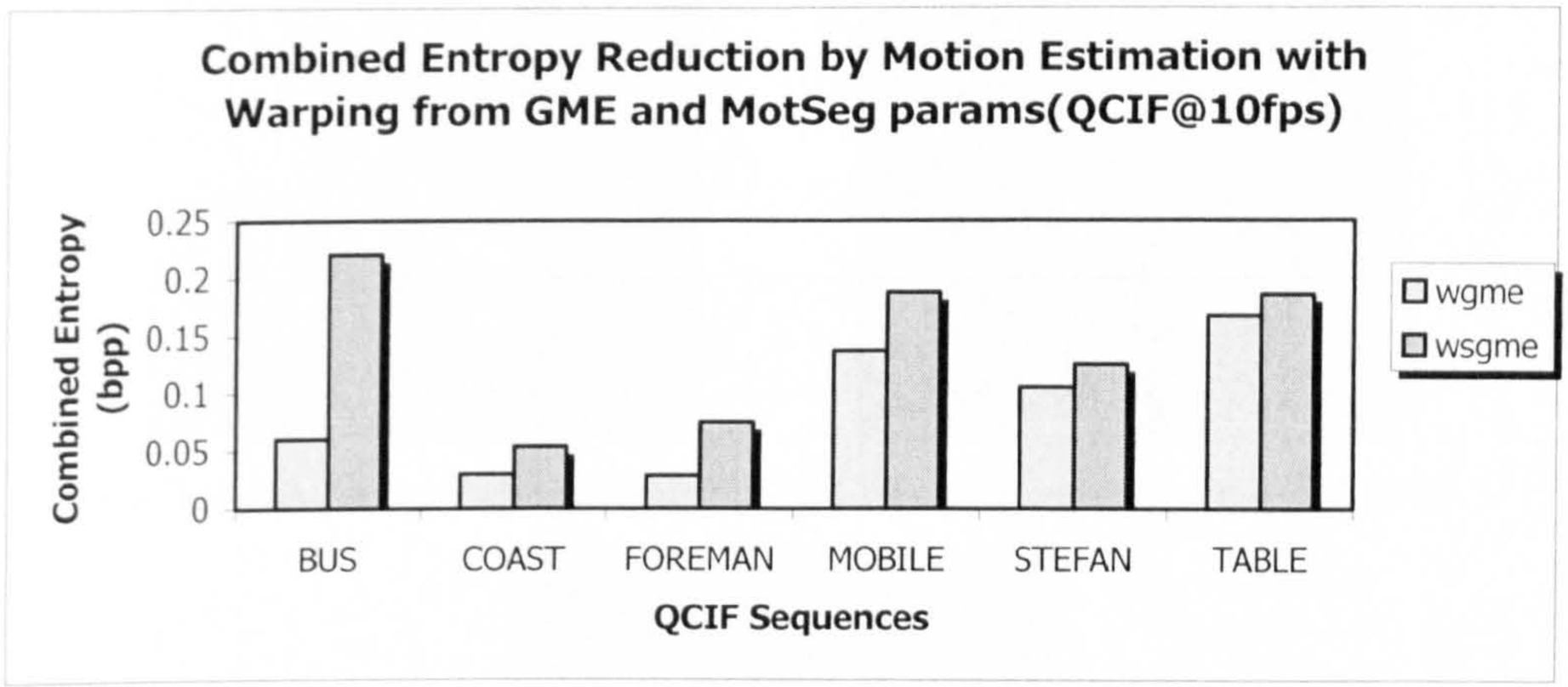
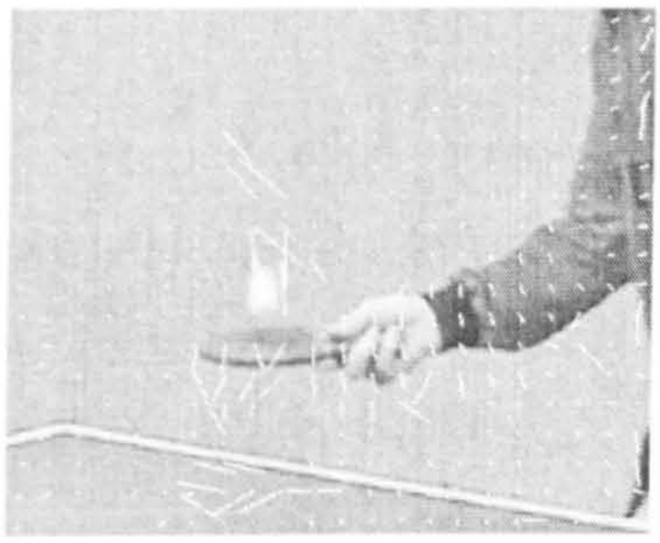
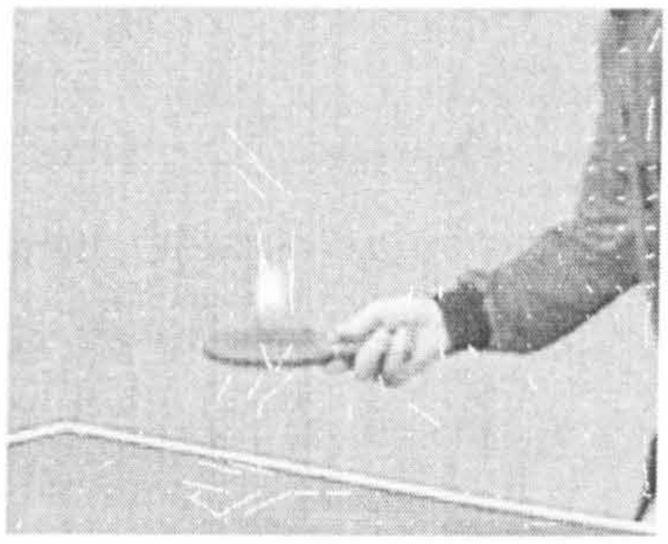


Figure 7.19. Charts showing the performance gained in terms of combined entropies of the motion vector field and residual texture residue by GME (PHGME) and MotSeg (PPAEMS).

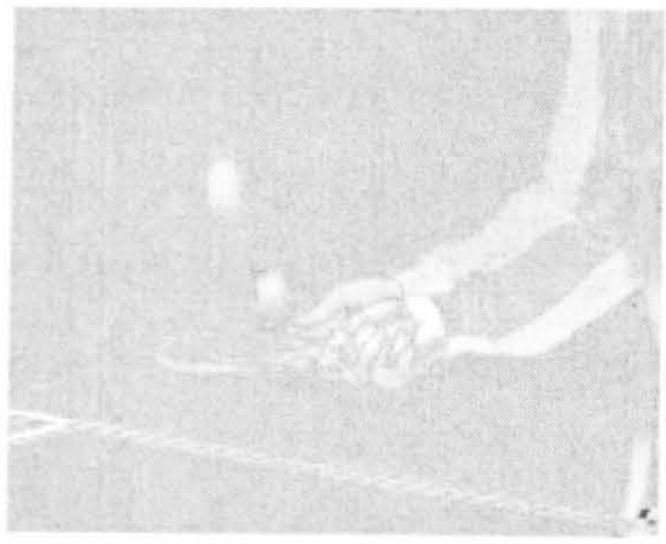
The comparison of the residual textures and vectors a sample frame in TABLE.QCIF is shown in Figure 7.20. Comparison between Panels (a) and (b) shows that PPAEMS produces less motion residues than PHGME, thus reducing motion entropy. Panel (c) shows that the multiply-warped reference frames reduce the texture entropies. The combined effect results in much lower entropy by PPAEMS, more than enough to compensate the overhead of that required for partition information in PPAEMS.



(a) Residual vectors after PHGME.



(b) Residual vectors after PPAEMS.



(c) Difference in Textural Residue Reduction

Figure 7.20. A comparison of the performance of warping with GME (PHGME) and MotSeg (PPAEMS) on TABLE.QCIF: (a) is the residual motion vector field after PHGME component is removed; (b) is residual motion vector field after PPAEMS; (c) is difference between the energy of the textural residues left after PPAEMS and PHGME – brighter regions represent better performance of SPHMS over SIRGME in terms of removing textural data with warping reference frame.

7.4 Conclusions and Recommendations

In this chapter, the advantages of global motion estimation are further exploited by finding multiple model parameters to separate regions of a frame. This is done via EM based segmentation on the QBMA motion vector field. Hough Transform-based segmentation initialization adds robustness to the estimation of initial parameters and enhances the chance of the EM algorithm reaching its global optimum point. A queue-based segment simplification generates a less fragmented segmentation map, thus providing a more natural segmentation and reduces the entropy of the differential segmentation map.

From Figure 7.18 and Figure 7.19, multiple motion segments representation can outperform that of global motion estimation in both providing a lower motion entropy and a better match in the reference frame by warping. The success of motion segmentation, however, is reduced due to the segmentation information accompanied with motion segmentation. This additional amount of information lessens the

edge of motion segmentation over global motion estimation, especially when the global motion dominates.

Hence, despite the great reduction in motion entropy, motion segmentation introduces the extra bit-budget – the segmentation map information. Unlike the motion parameters, which constitute a negligible portion of the overall entropy, the segmentation map is coded on a per-block basis and constitutes a substantial amount of information. This thesis does not explore the various possible means of coding this map efficiently. It is recommended that in the short future, research be made into various means of partition coding.

In any case, the proposed PPAEMS method can offer a reduction of 0.15 bpp for QCIF@10fps and 0.05 bpp for CIF@30fps, or 38 kbps and 150 kbps for sequences with more than one dominant moving object. In addition, the motion segmentation algorithm can be readily combined with any block-texture based segmentation algorithm to achieve a more natural joint motion-texture segmentation framework.

Chapter 8:

Conclusions and Future Work

8.1 Conclusions

This thesis has contributed novel algorithms in three major areas of digital video processing: local motion estimation, global motion estimation and motion segmentation for real-time applications. The work involved the adaptation of the block-matching algorithms using concepts derived from pixel-based algorithms. The basic structure of the SAD-map is introduced and used extensively in the work described. Sub-pixel resolution of the local motion field is obtained without interpolating the reference frames by modelling the SAD distribution about the local minimum. By appropriately processing the SAD-map, a novel reliability measure (MCS) is introduced which is shown empirically to be a better indicator of the confidence level of the motion vector within the block. A priority-queue-based algorithm is then proposed which makes use of the reliability measure as the priority, and is called the queue-based BMA (QBMA). QBMA subsequently imposes a smoothness constraint that results in a motion vector field having lower entropy while maintaining low residue energy. The interpolation-free sub-pixel modelling algorithm can bring about quality improvements of >75% of that obtainable from actual interpolation without the extra processing and memory requirements. The QBMA can bring about 12 kbps bit rate reduction by adapting a smoother motion vector field without increasing the residue entropy significantly.

QBMA and MCS are used in the proposed SIRGME global motion estimation to reduce the motion entropy further by representing the motion field as a single parameter set and residual vector field. The SIRGME algorithm is also shown to improve coding gain by warping the reference frame. In the sequences used (CIF@30fps and QCIF@10fps with typical target bitrates of 20 kbps to 9 Mbps), bit rate savings of 12-144 kbps were observed. Overall, average bit-rate savings between 0.1% and 1% can be obtained. It should be pointed out that the stated savings are information-theoretic and is not based on any simulations done using any specific video encoder. This is intentional by the author so as to provide an upper bound to the bit savings and not to peg the results to any standards which may not stand the trials of time. An alternative GME algorithm which is much more robust towards outliers based on Hough Transform is also proposed. Termed progressive Hough transform-based GME (PHGME), the novel algorithm introduces several adaptations to the standard Hough Transform to

reduce complexity such that PHGME is suitable for real-time application. A further 12-60 kbps savings can be obtained by using PHGME over SIRGME.

In sequences with more than one dominant moving region, GME can be outperformed by segmenting these regions, each with their own global motion parameters. The adaptive EM-based motion segmentation (AEMS) makes use of QBMA fields and the SAD-map to improve the accuracies of the segmentation results. AEMS is further augmented by Hough Transform-based segment initialization to improve convergence rate and to increase the chances of converging toward the global EM minimum. Several segment-simplification post-processing algorithms are also proposed to reduce the entropy of the segmentation map. With PPAEMS, 30-150 kbps of bit-rate savings can be achieved.

The simulations described in this thesis are carried out on an 850 MHz Pentium Laptop with 512 MB RAM. The simulation timings obtained show near real-time performance for QBMA, SIGME and PHGME (10-15 fps for QCIF and 0.5-3 fps for CIF); the real-time processing requirements of PPAEMS can yet be met with the simulation hardware. However, with the rate of advancements in processor and DSP and VHDL technologies, the author is confident that all the proposed algorithms can be implemented in real-time in the near future.

8.2 Future Work

The research work described in this thesis has been focused mainly on utilising the SAD-map concept in BMA to achieve higher video compression rates by reducing inter-frame redundancies. There are several issues yet to be tackled, and these provide a future work plan for the author.

An important issue regarding the smoothness constraint used in QBMA is how to make the constraint factor adaptive to different regions of the frame. Current implementations use a constant constraint value which provides a good compromise amongst the test sequences. However, much can be gained if the factor is made variable, perhaps according to how the reliability measures are related between neighbouring blocks. A more global approach can be adopted which will further improve the combined entropies of the vector field and textural residues. A candidate for this is the mean-shift algorithm.

Global motion estimation via the Hough Transform is extremely robust and can result in very accurate estimates. However the precision is limited due to quantization. Regression, on the other hand, can produce very high resolution. The downside is that the GME problem is highly multimodal and regression tends to be trapped in local optimum points. Future work will involve combining HT and regression to take advantage of the former's robustness and latter's precision.

The area of motion segmentation has been, and still is, an area of immense research. Optimization-related work is an especially important area of research due to increasing interest in surveillance and

monitoring applications. Joint texture and motion segmentation is an important area of research in which the author has great interest.

All the performance measurements made in this thesis were based on the hypothesis that bit-rate has a direct link with entropy. This premise *is yet to be tested with* an actual encoder system. The author, given enough funding, intends to implement all proposed work within an existing video coding standard such as H.264.

Due to the low complexity of the proposed QBMA, HT and EM algorithms on block-based motion estimation, the algorithm may be a good step for initializing pixel-based motion estimation and segmentation for video analysis applications requiring high accuracies. With proper funding, the algorithms can be extended to pixel-based applications like motion tracking and moving object extraction in pan-tilt-and-zoom cameras.

References

[Abd-92] I. M. Abdelqader, S. A. Rajala, W. E. Snyder, "Energy Minimization Approach to Motion Estimation," Sig. Proc., Aug. 1992.

[Ade-94] E. A. Adelson and J. Y. A. Wang, "Representing Moving Images with Layers," IEEE Trans. on IP, Sep. 1994, pp. 625-638.

[Adi-95] G. Adiv, "Determining Three-dimensional Motion and Structure from Optimal Flow Generated by Several Moving Objects," IEEE Trans. on PAMI, 1995, Vol. 7, pp 384-401.

[Alt-98] Y. Altunbasak, P.E. Eren, and A.M. Tekalp, "Region-based Parametric Motion Segmentation using Colour Information," Jnl. of GMIP, Jan. 1998, Vol. 60, pp. 13-23.

[Ava-00] O. Avaro, A. Eleftheriadis, et al, "MPEG-4 System: Overview," Sig. Proc.: Img. Comms., 2000, 15, pp. 281-298

[Bha-97] V. Bhaskaran, K. Konstantinides, "Image and Video Compression Standards, Algorithms and Architectures," Kluwer Academic Publishers, 2nd Ed. 1997.

[Ben-94] M. Ben-Ezra and S. Peleg and B. Rousso, "Motion Segmentation Using Convergence Properties," APRA Image Understanding Workshop, Nov. 1994.

[Bes-86] J. Besag, "On the Statistical Analysis of Dirty Images," Jnl. of RSS, Series B, Vol. 48, 1986, pp. 259-302.

[Bil-97] J. Bilmes, "A Gentle Tutorial on the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models," Technical Report, University of Berkeley, 1997, ICSI-TR-97-021.

[Bla-96] M. J. Black and P. Anandan, "The Robust Estimation of Multiple Motions: Parametric and Piecewise-smooth Flow Fields," CVIU, Jan. 1996, Vol. 63 No. 1, pp. 75-104.

[Bob-93] M. Bober and J. Kittler, "A Hough Transform based Hierarchical Algorithm for Motion Segmentation and Estimation," IEE Coll. Hough Transform, May 1993, Vol. 12, pp. 1-4.

[Bor-97] G. D. Borshukov, G. Bozdagi, Y. Altunbasak and A. M. Tekalp, "Motion Segmentation by Multistage Affine Classification," IEEE Trans. on IP, Nov. 1997, Vol. 6 No. 11, pp. 1591-1594.

[Bor-02] J. Bormans, K. Hill, "MPEG-21 Overview V.5," ISO/IEC/JTC1/SC29/WG11/N5231, October 2002.

[Bra-96] J. C. Brailean and A. K. Katsaggelos, "A Recursive Nonstationary MAP Displacement Vector Field Estimation Algorithm," ICIP, Sep. 1996, Vol. 1, pp. 917-920.

[Bru-01] M. Brünig and W. Niehsen, "Fast Full-Search Block Matching," IEEE Trans. on CSVT, February 2001, Volume 11, Number 2, pp. 241-247.

[Caf-83] C. Cafforio and F. Rocca, "The Differential method for Image Motion Estimation," in Image Sequence Processing and Dynamic Scene Analysis, 1983, pp. 104-124.

[Cha-97] M.M. Chang, A.M. Tekalp, and M.I. Sezan, "Simultaneous Motion Estimation and Segmentation," IEEE Trans. on IP, Sep. 1997, Vol. 6, Issue 9, pp. 1326-1333.

[Cho-90] P. B. Cho and C. M. Brown, "The Theory and Practice of Bayesian Image Labelling," Int. Jnl. on CV, Volume 4, 1990, pp 185-200.

[Cla-95] R. J. Clarke, "Digital Compression of Still Images and Video," Academic Press, 1995.

[Cot-98] M. G. Cote, Berna Erol and F. Kossentini, "H.263+: Video coding at Low Bit Rates," IEEE Trans. on CSVT, Vol. 8, No. 7, pp. 849-866, Nov. 1998.

[Cre-03] D. Cremers and A. Yuille, "A Generative Model Based Approach to Motion Segmentation," in German Conference on Pattern Recognition, Sep. 2003, Vol. 2781, pp. 313-320.

[Dah-01] R. Dahyot and P. Charbonnier, "Unsupervised Statistical Detection of Changing Objects in Camera-in-Motion Video," ICIP, Oct. 2001.

[Dan-03] A. Dante and M. Brookes, "Precise Real-time Outlier Removal from Motion Vector Fields for 3D Reconstruction," ICIP, Sep. 2003, Vol. 1, pp. 393-396.

[Duf-95a] F. Dufaux and F. Moscheni, "Motion Estimation Techniques for Digital TV: A Review and a New Contribution," Proceedings of IEEE, Jun. 1995, Vol. 83, No. 6, pp. 858-875.

[Duf-95b] F. Dufaux, F. Moscheni and A. Lippman, "Spatio-temporal Segmentation based on Motion and Static Segmentation," ICIP, Oct 1995, Vol. 1, pp. 306-309.

[Duf-00] F. Dufaux and J. Konrad, "Efficient, Robust, and Fast Global Motion Estimation for Video Coding," IEEE Trans. on IP., March 2000, Vol. 9, No. 3, pp. 497-501.

[Dur-00] E. Durucan and T. Ebrahimi, "Robust and Illumination Invariant Change Detection Based on Linear Dependence for Surveillance Application," EUSIPCO, Sep. 2000, pp. 1141-1144.

[Ebr-00] T. Ebrahimi and C. Horne, "MPEG-4 Natural Video Coding - An Overview," Sig. Proc.: Img. Comms., 2000, Vol. 15, pp. 365-385.

[Eis-91] Z. Eisips and D. Malah, "Global Motion Estimation for Image Sequence Coding Applications," in IEEE Convention in Israel, May 91, pp. 186-189.

[For-02] H. Foroosh, J. B. Zerubia, and M. Berthod, "Extension of Phase Correlation to Subpixel Registration," IEEE Trans. on IP, Mar. 2002, Vol. 11, No. 3, pp. 188-200.

[Gem-84] S. Geman and D. Geman, "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images," IEEE Trans. on PAMI, Vol. 6, Nov. 1984, pp. 721-741.

[Giu-99] G. Giunta, U. Mascia, "Estimation of Global Motion Parameters by Complex Linear Regression," IEEE Trans. on IP, Nov. 1999, Vol. 8, Issue 11, pp. 1652-1657.

[Has-94] B.G. Haskell, P.G. Howard, Y.A. LeCun, A. Puri, J. Ostermann, M.R. Civanlar, L. Rabiner, L. Bottou, P. Haffner, "Image and Video Coding - emerging Standards and Beyond," IEEE Trans. on CSVT, Nov. 1998, Vol. 8, Issue 7, pp. 814-837.

[He-01] Y. He, B. Feng, S. Yang and Y. Zhong, "Fast Global Motion Estimation for Global Motion Compensation Coding," ISCAS, May. 2001, Vol. 2, pp. 233-236.

[Hei-90] F. Heitz, and P. Bouthemy, "Motion Estimation and Segmentation using a Global Bayesian Approach," ICASSP, Apr. 1990, Vol. 4, pp. 2305-2308.

[Heu-99] J. Heuer and A. Kaup, "Global Motion Estimation in Image Sequences Using Robust Motion Vector Field Segmentation," in Association for Computing Machinery Multimedia Conference, Nov. 1999, pp. 261-264.

[Hil-99] L. Hill and T. Vlachos, "On the Estimation of Global Motion using Phase Correlation for Broadcast," IPA, Jul. 1999, pp. 721-725.

[Hil-01] L. Hill and T. Vlachos, "Fast Motion Estimation using a Reliability Weighted Robust Search," IEE Electronic Letters, 2001, Vol. 37, No. 7, pp. 418-420.

[Hor-81] B. K. P. Horn and B. G. Schunck, "Determining Optical Flow," Artificial Intelligent, 1981, Volume 17, Number 1-3, pp. 185-203.

[Hua-02] Y. W. Huang, S. Y. Chien, B. Y. Hsieh, and L. G. Chen, "Automatic Threshold Decision of Background Registration Technique for Video Segmentation," VCIP, Jan. 2002.

[Iai-81] J. Iain, A. Jain, "Displacement Measurement and Its Application in Interframe Image Coding," IEEE Trans. on COMMS, Dec. 1981, Vol. 29, Issue 12, pp. 1799-1808.

[Ira-92] M. Irani and S. Peleg, "Image Sequence Enhancement Using Multiple Motions Analysis," CVPR, Jun 1992, pp. 216-221.

[ISO-98] ISO, "Information technology – Generic coding of audio-visual objects – Part 2: Visual," October 1998.

[ITU-93] ITU-T, "H.261: Video Codec for Audiovisual Services at p x 64 kbit/s," Mar. 1993.

[ITU-98] ITU-T, "H.263: Video coding for low bit rate communication," Feb 1998.

[Iu-93] S.L. Iu, "Robust Estimation of Motion Vector Fields with Discontinuity and Occlusion using Local Outliers Rejection," Jnl. of VCIR, 1993, Vol. 2094, pp. 588-599.

[Jin-00] K. Jinzenji, S. Okada, H. Watanabe, N. Kobayashi, "Automatic Two-layer Video Object Plane Generation Scheme and its application to MPEG-4 Video Coding," ISCAS, May. 2000, Vol 3, pp. 606-609.

[Joz-97] H. Jozawa, K. Kamikura, A. Sagata, H. Kotera and H. Watanabe, "Two-stage Motion Compensation using Adaptive Global MC and Local Affine MC," IEEE Trans. on CSVT, Feb. 1997, Vol. 7, No. 1, pp. 75-85.

[JVT-02] Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, "DRAFT ISO/IEC 14496-10 : 2002 (E) DRAFT ITU-T Rec. H.264 (2002 E)," JVT-D157, Aug. 2002.

[Kal-96] H. Kalviainen, "Motion Detection using the Randomised Hough Transform: Exploiting Gradient Information and Detecting Multiple Moving Objects," VISP, Dec. 1996, Issue 6, pp. 361-369.

[Kel-03] Y. Keller and A. Averbuch, "Fast Gradient Methods based on Global Motion Estimation for Video Compression," IEEE Trans. on CSVT, Apr. 2003, Vol. 13, No. 4, pp. 300-309.

[Kim-99a] E. T. Kim and H. M. Kim, "Fast and Robust Parameter Estimation Method for Global Motion Compensation in the Video Coder," IEEE Trans. on CE, Feb. 1999, Vol. 45, No. 1, pp. 76-83.

[Kim-99b] C. Kim, J. Hwang, "A Fast and Robust Moving Object Segmentation in Video Sequences," ICIP, Oct. 1999, Vol. 2, pp. 131-134.

[Kir-83] S. Kirkpatrick, C. D. Gelatt, Jr. M. P. Vecchi, "Optimization by Simulated Annealing," Science, May 1983, Vol. 220, No. 4598, pp. 671-680.

[Koc-96] U. V. Koc and K. J. R. Liu, "DCT-based Subpixel Motion Estimation," ICASSP, Apr. 1996, Vol. 4, pp. 1931-1934.

[Koc-98] U. Koc, K.J.R. Liu, "Interpolation-free Subpixel Motion Estimation Techniques in DCT Domain," IEEE Trans. on CSVT, Aug. 1998, Vol. 8, Issue 4, pp. 460-487.

[Koe-00] R. Koenen, F. Pereira, "MPEG-7: A Standardised Description of Audiovisual Content," Sig. Proc.: Img. Comms., 2000, 16, pp. 5-13.

[Kog-81] T. Koga, K. Iinuma, A. Hirano and T. Ishinguro, "Motion-compensated Interframe Coding for Video Conferencing," National Telecommunications Conference, 1981, pp. G5.3.1-G5.3.5.

[Kon-92] J. Konrad and E. Dubois, "Bayesian Estimation of Motion Vector Fields," IEEE Trans. on PAMI, Sep. 1992, Vol. 4, No. 9, pp. 910-927.

[Kuo-98] T. Kuo, C.C.J. Kuo, "Fast Overlapped Block Motion Compensation with Checkerboard Block Partitioning," IEEE Trans. on CSVT, Oct. 1998, Vol. 8, No. 6, pp. 705-712.

[Lai-77] N. Laid, A. Dempster and D. Rubin, "Maximum-likelihood from Incomplete Data via the EM Algorithm," Jnl. of RSS, Series B, 1977, Vol. 39, pp 1-38.

[Lee-97] M. Lee, W. Chen, C.B. Lin, C. Gu, T. Markoc, S.I. Zabinsky and R. Szeliski, "A Layered Video Object Coding System using Sprite and Affine Motion Model," IEEE Trans. Circuits Syst. Video Technol., Feb. 1999, Vol. 7, Issue 1, pp. 130-145.

[Lee-99] S. Lee, J. Kim, "Fast Block Motion Estimation for Overlapped Motion Compensation Using Selective Pixel Matching," ICIP, Oct. 1999, pp. 80-83.

[Leo-99] T. Leonard and S. J. Hsu, "Bayesian Methods," Cambridge University Press, 1999.

[Li-94] R. Li, B. Zeng, M. L. Liou, "A New Three-step Search Algorithm for Block Motion Estimation," IEEE Trans. on CSVT, August 1994, Volume 4, pp. 438-442.

[Li-00] S. Z. Li, "Modeling Image Analysis Problems Using Markov Random Fields," Handbook of Statistics, 2000, Vol. 20, pp. 1-43.

[Li-01] H. Li, B.J. Tye, E.P. Ong, W.S. Lin and C.C. Ko, "Multiple Motion Object Segmentation based on Homogenous Region Merging," ISCAS, May. 2001, Vol. 5, pp. 175-178.

[Lin-99] C. T. Lin, S. C. Hsiao and G. D. Wu, "New Techniques on Deformed Image Motion Estimation and Compensation," IEEE Trans. on SMC, Dec. 1999, Vol. 29, No. 6, pp. 846-859.

[Liu-92] S. Liu, M. Hayes, "Segmentation-based Coding of Motion Difference and Motion Field Images for Low Bit-rate Video Compression," ICASSP, Mar. 1992, Vol. 3, pp. 525-528.

[Liu-96] L. Liu and E. Feig, "A Block-Based Gradient Descent Search Algorithm for Block Motion Estimation Video Coding," IEEE Trans. on CSVT, August 1996, Volume 6, Number 4.

[Mar-02] J. M. Martínez, "MPEG-7 Overview V.8," ISO/IEC/JTC1/SC29/WG11/N5231, July 2002.

[Mei-97] T. Meier, K.N. Ngan, G. Crebbin, "A robust Markovian segmentation based on highest confidence first (HCF)," ICIP, Oct 1997, Vol.1, p. 261.

[Mos-95] F. Moscheni, F. Dufaux, M. Kunt, "A New Two-stage Global/Local Motion Estimation based on a Background/Foreground Segmentation," ICASSP, May. 1995, Vol. 4, pp. 2261-2264.

[MPE-93] MPEG-1 Video Group, "Information Technology – Coding of Moving Pictures and associated Audio for Digital Storage Media up to about 1.5 Mbits/s: Part 2 – Video," ISO/IEC 11172-2, International Standard, 1993

[MPE-95] MPEG-2 Video Group, "Generic Coding of Moving Pictures and associated Audio: Part 2 – Video," ISO/IEC 13818-2, International Standard, 1995

[Mur-87] D. W. Murray and B. F. Buxton, "Scene Segmentation from Visual Motion using Global Optimization," IEEE Trans. on PAMI, Mar. 1987, Vol. 2, pp. 220-228.

[Nag-86] H. H, Nagel and W. Enkelmann, "An Investigation of Smoothness Constraints for the Estimation of Displacement Vector Fileds from Image Sequences," IEEE Trans. on PAMI, Volume 8, pp 565-593, 1986.

[Net-79] A. N. Netravali and J. D. Robbins, "Motion Compensated Television Coding: Part 1, " Bell System Technical Journal, Vol. 58, Mar 1979, pp 631-670

[Ngu-00] H. T. Nguyen, M. Worring, A. Dev, "Detection of Moving Objects in Video Using a Robust Motion Similarity Measure," IEEE Trans. on IP, Jan. 2000, Vol. 9, No. 1, pp. 137-141.

[Nic-91] H. Nicolas and C. Labit, "Global Motion Identification for Image Sequence Analysis and Coding," ICASSP, May. 1991, Vol. 4, pp. 2825-2828.

[Nit-00] S. Nitsuwat, J. S. Jin, and N. M. Hudson, "Motion-based Video Segmentation using Fuzzy Clustering and Classical Mixture Model," ICIP, Sep. 2000, Vol. 1, pp. 300-303.

[Orc-94] M. T. Orchard and G. J. Sullivan, "Overlapped block motion compensation: An estimation-theoretic approach," IEEE Trans. on IP, Sep. 1994, Vol. 3, pp. 693-699

[Pan-02] K. Panusopone and D. M. Baylon, "An Analysis and Efficient Implementation of Half-Pel Motion Estimation," IEEE Trans. on CSVT, Aug. 2002, Vol. 12, No. 8, pp. 724-729.

[Par-94] J. Park, N. Yagi, K. Enami, K. Aizawa and M. Hatori, "Estimation of Camera Parameters from Image Sequence for Model-based Video Coding," IEEE Trans. on CSVT, Jun. 1994, Vol. 4, Iss. 3, pp. 288-296.

[Pat-02] I. Patras, E. A. Hendriks and R. L. Lagendijk, "Confidence Measures for Block Matching Motion Estimation," ICIP, Sep. 2002, Vol. 2, pp. 277-280.

[Pel-90] S. Peleg and H. Rom, "Motion Based Segmentation," ICPR, Jun 1990, Vol. 1, pp. 109-113.

[Per-00] F. Pereira, "MPEG-4: Why, what, how and when?" Sig. Proc.: Img. Comms., 2000, Vol. 15, pp. 271-279.

[Pre-02] W. H. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery, "Minimization or Maximization of Functions," "Numerical Recipes in C++," Cambridge University Press, 2002, Chapter 10, pp. 398-460.

[Rat-99] G. B. Rath and A. Makur, "Iterative Least Squares and Compression Based Estimations for a Four-Parameter Linear Global Motion Model and Global Motion Compensation," IEEE Trans. on CSVT, Oct. 1999, Vol. 9, No. 7, pp. 1075-1099.

[Rei-97] M. M. Reid, R. J. Millar and N. D. Black, "Second-generation Image Coding: an Overview," ACM Computing Surveys, Mar. 1997, Vol. 29, Issue 1, pp. 3-29.

[Ric-www] I. E. G. Richardson, "H.264 / MPEG-4 Part 10 White Paper," http://www.vcodex.com

[Rij-96] K. J. Rijkse, "H.263: Video Coding for Low-bit-Rate Communication," IEEE Comms. Mag., Dec 1996.

[Rob-01] A. Robles-Kelly and E. R. Hancock, "An EM-like Algorithm for Motion Segmentation via Eigendecomposition," BMVC, 2001, pp. 654-661.

[Ros-98] K. Rose. "Deterministic Annealing for Clustering, Compression, Clustering, Compression, Classification, Regression, and Related Optimization Problems," Proceedings of the IEEE, Nov. 1998, Vol. 86, No. 11, pp. 2210-2239.

[Saw-95] H. S. Sawhney, S. Ayer and M. Gorkani, "Model-based 2D&3D Dominant Motion Estimation for Mosaicing and Video Representation," ICIP, Jun. 1995, pp. 583-590.

[Saw-96] H. S. Sawhney and S. Ayer, "Compact Representations of Videos Through Dominant and Multiple Motion Estimation," IEEE Trans. on PAMI, Aug. 1996, Vol. 18 , Issue 8, pp. pp. 814-830.

[Sch-96] M. Schutz and T. Ebrahimi, "Matching Error Based Criterion of Region Merging for Joint Motion Estimation and Segmentation Techniques," ICIP, 1996, pp. 509-512.

[Sha-48] C. E. Shannon, "A Mathematical Theory of Communication," Bell System Technical Journal, Vol. 27, Jul., Oct. 1948, pp. 379-423, 623-656.

[Sik-97] T. Sikora, "MPEG-4 Very Low Bit Rate Video," ISCAS, June 1997.

[Sil-98] M. Silveira, M. Piedade, "Joint Segmentation and Motion Estimation," ICIP, Oct. 1998, Vol. 2, pp. 657-661.

[Sil-01] M. Silveira and M. Piedade, "MRF-motion Segmentation Based on Dominant Motion Estimation and the Detection of Uncovered Regions," ICIP, Oct. 2001, Vol. 1, pp. 373-376.

[Smo-99] A. Smolic, T. Sikora, J.-R. Ohm, "Long-term Global Motion Estimation and its Application for Sprite Coding, Content Description, and Segmentation," IEEE Trans. on CSVT, Dec. 1999, Vol. 8, Iss. 9, pp. 1227-1242.

[Smo-00a] A. Smolic, M. Hoeynck and J. R. Ohm, "Low-complexity Global Motion Estimation from P-frame Motion Vectors for MPEG-7 Applications," ICIP, Sep. 2000, Vol. 2, pp. 271-274.

[Smo-00b] A. Smolic, and J. R. Ohm, "Robust Global Motion Estimation Using a Simplified M-estimator Approach," ICIP, Sep. 2000.

[Sri-85] R. Srinivasan and K. Rao, "Predictive Coding based on Efficient Motion Estimation," IEEE Trans. on COMMS, Aug. 1985, Vol. 33, pp. 888-896.

[Ste-99] E. Steinbach, T. Wiegand and B. Girod, "Using Multiple Global Motion Models for Improved Block-based Video Coding," ICIP, Oct. 1999, Vol. 2, pp. 56-60.

[Sti-94] C. Stiller, "Object-Oriented Video Coding Employing Dense Motion Fields," ICASSP, Apr. 1994, Vol. 5, pp. 273—276.

[Stu-92] K. W. Stuhlmuller and B. Girod, "Motion Segmentation for Region-based Coding," IPA, Jul. 1997, Vol. 2, pp. 650-654.

[Tao-97] B. Tao, M.T. Orchard, B. W. Dickinson, "Joint Application of Overlapped Block Motion Compensation and Loop Filtering for Low Bit-rate Video Coding," ICIP, Oct. 1997, Vol. 3, pp. 626-629.

[Tek-95] A. M. Tekalp, "Digital Video Processing," Prentice Hall, 1995

[Tia-95] T. Y. Tian and M. Shah, "Recovering 3D Motion of Multiple Objects using Adaptive Hough Transform," ICCV, Jun. 1995, pp. 284-289.

[Tri-01] B. Triggs, "Empirical Filter Estimation for Subpixel Interpolation and Matching," ICCV, 2001, Vol. 2, pp. 550-557.

[Tse-91] Y. T. Tse and R. L. Baker, "Global Zoom/Pan Estimation and Compensation for Video Compression," ICASSP, Apr. 1991, Vol. 4, pp. 2725-2728.

[Vas-01] N. Vasconcelos, A. Lippman, "Empirical Bayesian Motion Segmentation," IEEE Trans. on PAMI, Feb. 2001, Vol 23, Issue 2, pp. 217-221.

[Wal-84] D. Walker and K. Rao, "Improved Pel-Recursive Motion Compensation," IEEE Trans. on COMMS., Oct. 1984, Vol. 32, No. 10, pp. 1128-1134.

[Wan-97] D. Wang and L. Wang, "Global Motion Parameters Estimation using a Fast and Robust Algorithm," IEEE Trans. on CSVT, Oct. 1997, Vol. 7, Issue 5, pp. 823-826.

[Wan-00] R. Wang, H. Zhang and Y. Zhang, "A Confidence Measure Based Moving Object Extraction System Built for Compressed Domain," ISCAS, May. 2000, Vol. 5, pp. 21-24.

[Wan-04] Z. Wang, L. Lu and A. C. Bovik, "Video Quality Assessment Based on Structural Distortion Measurement," Sig. Proc.: Img. Comms., Jan 2004, Vol. 19 No. 1, pp. 1-9.

[Wei-96] Y. Weiss and E. H. Adelson, "A Unified Mixture Framework for Motion Segmentation: Incorporating Spatial Coherence and Estimating the Number of Models," CVPR, 1996, pp. 321-326.

[Wei-01] Y. Weiss, "Motion Segmentation using EM - a Short Tutorial," http://persci.mit.edu/2001.

[Wei-03] T. Wiegand, G. J. Sullivan, G. Bjntegaard and A. Luthra, "Overview of the H.264/AVC video coding standard," IEEE Trans. on CSVT, Jul. 2003, Vol. 13 Issue 7, pp. 560-576.

[Xio-97] Z. Xiong, T. Chiang and Y. Zhang, "Global Motion Compensation for Low Bitrate Video Coding," MMSP, Jun. 1997, pp. 195-200.

[Yos-97] T. Yoshida, H. Katoh, and Y. Sakai, "Block Matching Motion Estimation Using Block Integration Based on Reliability Metric," ICIP, Oct. 1997, Vol. 2, pp. 152-155.

[Zha-93] J. Zhang and J. Hanauer, "The Mean Field Theory for Image Motion Estimation," ICASSP, Apr. 1993, Vol. 5, pp. 197-200.

[Zha-97] K. Zhang, M. Bober and J. Kittler, "Image Sequence Coding Using Multiple-Level Segmentation and Affine Motion Estimation," IEEE Jnl. on SAC, Dec. 1997, Vol. 15, No. 9, pp. 1704-1713

[Zha-98] K. Zhang and J. Kittler, "Global Motion Estimation and Robust Regression for Video Coding," ICASSP, May. 1998, Vol. 5, pp. 2589-2592.

List of Abbreviations to Conferences and Publications

BMVC	British Machine Vision Conference
CVIU	Computer Vision and Image Understanding
CVPR	Computer Vision and Pattern Recognition
EUSIPCO	European Signal Processing Conference
ICASSP	International Conference on Acoustics, Speech and Signal Processing
ICCV	International Conference on Computer Vision
ICIP	International Conference on Image Processing
ICPR	International Conference on Pattern Recognition
IPA	International Conference on Image Processing and its Applications
ISCAS	International Symposium on Circuits and Systems
IEEE Comms. Mag.	IEEE Communications Magazine
IEEE Jnl. on SAC	IEEE Journal on Selected Areas of Communications
IEEE Trans. on CE	IEEE Transactions on Consumer Electronics
IEEE Trans. on COMMS	IEEE Transaction on Communications
IEEE Trans. on CSVT	IEEE Transaction on Circuits and System for Video Technology
IEEE Trans. on IP	IEEE Transaction on Image Processing
IEEE Trans. on PAMI	IEEE Transaction on Pattern Analysis and Machine Intelligence
IEEE Trans. on SMC	IEEE Transaction on Systems, Man and Cybernetics
Int. Jnl. on CV	International Journal on Computer Vision
Jnl. of GMIP	Journal of Graphical Models and Image Processing
Jnl. of RSS	Journal of Royal Statistics Society

Jnl. of VCIR	Journal of Visual Communication and Image Representation
MMSP	International Workshop on Multimedia Signal Processing
Sig. Proc.	Signal Processing
Sig. Proc.: Img. Comms.	Signal Processing: Image Communications
VCIP	SPIE Conference on Visual Communications and Image Processing
VISP	IEE Proceedings of Vision, Image and Signal Processing